EMBEDDIA

Cross-Lingual Embeddings for Less-Represented Languages in European News Media

Research and Innovation Action
Call: H2020-ICT-2018-1
Call topic: ICT-29-2018 A multilingual Next generation Internet
Project start: 1 January 2019 Project duration: 36 months

D2.6: Final multilingual keyword extraction techniques (T2.2)

Executive summary
In Task T2.2 "Multilingual keyword extraction and matching" we developed methods for extracting topical terms and keywords from the input text in a monolingual and multilingual problem setting, aiming to detect keywords and terms applicable in news linking, cross-lingual topic modelling and news analysis in later stages of the EMBEDDIA project. This deliverable describes the final keyword extraction techniques. The main contribution is a novel neural keyword identification method TNT-KID, which demonstrates excellent results on several benchmark datasets. We apply our method to a variety of media datasets, and also produce a combined system using TNT-KID and TF-IDF based methods to improve the recall. In addition, we present experiments in term-extraction and keyword alignment. Finally, we present selected experiments related to the identification of important terms for knowledge discovery.

Partner in charge: JSI

Project co-funded by the European Commission within Horizon 2020 Dissemination Level		
PU	Public	PU
PP	Restricted to other programme participants (including the Commission Services)	–
RE	Restricted to a group specified by the Consortium (including the Commission Services)	–
CO	Confidential, only for members of the Consortium (including the Commission Services)	–

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825153

Deliverable Information

Document administrative information	
Project acronym:	**EMBEDDIA**
Project number:	**825153**
Deliverable number:	**D2.6**
Deliverable full title:	**Final multilingual keyword extraction techniques**
Deliverable short title:	**Final keyword extraction**
Document identifier:	**EMBEDDIA-D26-FinalKeywordExtraction-T22-submitted**
Lead partner short name:	**JSI**
Report version:	**submitted**
Report submission date:	**31/12/2020**
Dissemination level:	**PU**
Nature:	**R = Report**
Lead author(s):	**Senja Pollak (JSI), Matej Martinc (JSI)**
Co-author(s):	**Andraž Repar (JSI), Blaž Škrlj (JSI), Boshko Koloski (JSI), Matej Ulčar (UL), Silver Traat (TEXTA)**
Status:	**__ draft, __ final, _x_ submitted**

The EMBEDDIA Consortium partner responsible for this deliverable has addressed all comments received. Changes to this document are detailed in the change log table below.

Change log

Date	Version number	Author/Editor	Summary of changes made
01/11/2020	v0.1	Senja Pollak (JSI)	Structure of the report.
01/11/2020	v0.2	Matej Martinc, Boshko Koloski, Andraž Repar, Blaž Škrlj, Senja Pollak (JSI)	First draft.
10/12/2020	v0.3	Senja Pollak (JSI)	Final for the internal review.
15/12/2020	v0.4	Marko Robnik-Sikonja (UL)	Internal review.
20/12/2020	v0.5	Hannu Toivonen (UH-CS)	Internal review.
22/12/2020	v0.6	Senja Pollak, Matej Martin, Andraž Repar (JSI)	Changes addressing reviewers' comments.
25/12/2020	prefinal	Nada Lavrač (JSI)	Report quality check.
26/12/2020	final	Senja Pollak (JSI)	Minor corrections.
29/12/2020	submitted	Tina Anžič (JSI)	Report submitted.

Table of Contents

List of abbreviations

BILSTM	Bidirectional Long Short-Term Memory
D	Deliverable
CRF	Conditional Random Fields
DoA	Description of Action
LBD	Literature-based Discovery
LSTM	Long Short Term Memory
NLL	Negative log-loss
T	Task
TNT-KID	Transformer-based Neural Tagger for Keyword Identification
TF-IDF	Term frequency - inverse document frequency

1 Introduction

The EMBEDDIA project aims to develop monolingual and cross-lingual technology for news media industry. The overall objective of WP2, named *Advanced NLP Technologies for Less-Resourced Languages*, is to advance a range of fundamental NLP technologies with the following specific objectives:

- Advance cross-lingual semantic enrichment, including named entity recognition and disambiguation, and event detection (inT2.1).
- Advance multilingual keyword and terminology extraction and matching technology (in T2.2).
- Advance multilingual natural language generation (in T2.3).

This deliverable (D2.6), entitled *Final multilingual keyword extraction techniques*, is the final report of T2.2, which started in M1 and lasted until M24, and presents the advances since the previous deliverable on Initial keyword extraction techniques (D2.3).

First we provide the definition of core terms used in this deliverable that were for the most part introduced already in deliverable D2.3.

Keywords are terms (i.e. expressions) that best describe the subject of a document (Beliga et al., 2015) and a good keyword effectively summarises the content of a document allowing it to be efficiently retrieved when needed.

Terms are verbal designations of general concepts in a specific subject field (as defined in ISO 1087 standard). In contrast to keywords, which are usually assigned on a single document level, terms are more frequently used on a document collection level, i.e. domain level. While not all terms are keywords, there is a strong overlap between the most frequent terms and keywords, therefore term extraction techniques can be applied successfully to keyword extraction.

Keyword extraction refers to the process of automatically extracting keywords from documents.

Term alignment (or terminology alignment) is the process of aligning terms between two candidate term lists in two languages.

Keywords are the most important words describing a document. In a media analysis setting, keywords correspond to tags that are added to articles by news providers and support search on the news portals and link articles together. During the project, the media partners—more specifically ExM (Ekspress meedia) and 24sata for whom Trikoder (TRI) provides services—have clearly identified keyword extraction as one of the tasks where automatization is feasible and can clearly benefit in real life settings. On the other hand, *terms*, which denote expressions characteristic for a specific domain, are most frequently discussed in the fields of translation and terminology management. However, in media analysis, terms are relevant for analysing the vocabulary of different news categories (e.g., sports vs. foreign policy). As similar techniques can be used for both, the field has a strong exploitation potential for applying methods from the media setting (keywords) to the translation industry and terminography (terms), and vice versa. *Term alignment*, referring to alignment of terms across two or more languages, is relevant also in the news analysis setting when used for finding corresponding keywords from tagsets in different languages.

In this report, we present our supervised keyword extraction system TNT-KID (Martinc, Škrlj, & Pollak, 2020). Its initial version was presented in D2.3, but we improved the methods and performed experiments during the second year of the project and here present the final method and results. The method was applied to media partners' datasets, and a combined method complementing TNT-KID results with tagset matching techniques was proposed to satisfy the need of media partners to identify a larger number of keywords. Next, we briefly present experiments by TEXTA, where keyword extraction is considered as a classification task (this approach named TEXTA Hybrid tagger is done in collaboration with WP6, see deliverables D6.7 and D6.8). Further, in this report, we present how term alignment techniques presented in the previous deliverable D2.3 were applied to the ExM tagset for aligning keywords

in Estonian and Russian, and present also further developments in term extraction and alignment methods using embeddings.

In addition, this report presents also the advances in keyword and term extraction applied to scientific texts, which is a relevant application area especially given the time of the COVID-19 pandemics. The work includes the presentation of how our unsupervised keyword extraction method RaKUn (Škrlj et al., 2019) (that was presented in the previous deliverable of this task D2.3) was used to build the COVID-19 Explorer (`http://covid19explorer.ijs.si/`), a tool for fast and interactive literature prioritization. Next, we present two methods for identification of terms by which new scientific knowledge can be discovered from scientific literature, with applications in the biology and COVID-19 domains. This work shows promising exploitation potential of the results of the project.

The work performed in task T2.2 resulted in several papers, which are included in the Appendices of this deliverable:

- **Martinc, Škrlj, & Pollak (2020)**: "TNT-KID: Transformer-based Neural Tagger for Keyword Identification", submitted to Natural language engineering journal (2nd revision), available as preprint (Appendix A), introduces our novel transformer-based keyword extraction approach (covered in Section 2);
- **Lavrač et al. (2020)**: "Bisociative Literature-Based Discovery: Lessons Learned and New Word Embedding Approach", published in New generation computing journal (Appendix B), presenting a bisociative scientific knowledge discovery approach using word embeddings (summarised in Section 7.3);
- **Martinc, Škrlj, et al. (2020)**: "COVID-19 therapy target discovery with context-aware literature mining", published in Proceedings of Discovery science conference (Appendix C), presenting a contextual-embeddings-based term identification for scientific discovery (introduced in Section 7.2);
- **Grčić-Simeunović et al. (2020 (to apprear))**: A bilingual approach to specialised adjectives through word embeddings in the karstology domain, to be published in proceedings of the TOTH conference (Appendix D), which presents a study on embeddings-based terminological study on adjectives (summarised in Section 6.3).
- **Covid-19 explorer** (`http://covid19explorer.ijs.si/`): a tool for prioritization of scientific literature of Covid-19 based on research developed in the first part of T2.2 (Škrlj et al. (2019)) (see Section 7.1).

This deliverable is structured as follows. In Section 2, we present the core part of the work presented in this deliverable, the design and evaluation of the supervised neural keyword extractor TNT-KID. In Section 3 we focus on media partners' dataset on which we apply TNT-KID, and a development of a combined approach for improving a recall. In Section 4, we briefly present the work on keyword extraction on a media dataset (from external media company), where the keyword matching is framed as a classification task. In Section 5, we present the application of term alignment techniques to the ExM tagset (keywords of our media partner contain Estonian and Russian keywords) in and further advances on the term alignment method, followed by novel term extraction experiments in Section 6. We finish the report by describing our work on terms and keywords in scientific papers (Section 7), which is showing exploitation potential of EMBEDDIA developments. The two final sections present conclusions and future work (in Section 8) and the outputs resulting from our work in the final year of the task T2.2 (Section 9).

2 TNT-KID: Novel neural keyword extraction method and its evaluation on public datasets

This section presents a novel keyword extraction method Transformer-based Neural Tagger for Keyword IDentification (TNT-KID). In the previous deliverable on Initial keyword extraction techniques (D2.3), we

have already introduced initial developments of our proposed method TNT-KID. Since then, we have made many advances, as we improved the method (e.g., adapting the attention mechanism and loss function), added novel experiments (new datasets and state-of-the-art approaches for comparison), added ablation studies, visualisation and wrote the paper Martinc, Škrlj, & Pollak (2020) (presented in Appendix A), which was submitted to the Natural Language Engineering journal.

Unsupervised approaches, such as RaKUn (Škrlj et al., 2019) and YAKE (Campos et al., 2018b), work fairly well and have some advantages over supervised approaches, as they are language and genre independent, do not require any training, and are computationally undemanding. Nevertheless, as already explained in previous deliverable D2.3, the motivation for the novel supervised keyword extraction method comes from the observation that unsupervised approaches also have a couple of crucial deficiencies:

- The above unsupervised approaches use simple statistics like word occurrence and co-occurrence (as in term frequency – inverse document frequency, TF-IDF, and graph based features, such as PageRank) to detect the importance of each word in the document. They are therefore semantically relatively weak.
- Since these systems cannot be trained on a dataset with manually labelled keywords, they cannot be adapted to the specifics of the syntax, semantics, content, genre and keyword assignment regime of a specific text (e.g., a variance in a number of keywords).

These deficiencies result in a much worse performance when compared to the state-of-the-art supervised algorithms (see Table 2), which have a direct access to the gold standard keyword set for each text during the training phase, enabling more efficient adaptation. The newest supervised neural algorithms (Meng et al., 2019; Yuan et al., 2019) therefore achieve excellent performance under satisfactory training conditions and can model semantic relations much more efficiently than algorithms based on simpler word frequency statistics. On the other hand, these algorithms are resource demanding, require vast amount of domain specific data for training and can therefore not be used in domains and languages that lack manually labelled resources of sufficient size.

TNT-KID[1] is capable of overcoming the aforementioned deficiencies of supervised and unsupervised approaches. We show that, while requiring only a fraction of manually labelled data required by other neural approaches, the proposed approach achieves the performance comparable to the state-of-the-art supervised approaches on test sets for which a lot of manually labelled training data is available. On the other hand, if training data that is sufficiently similar to the test data is scarce, our model outperforms the state-of-the-art approaches by a large margin. This is achieved by leveraging the transfer learning technique, where a keyword tagger is first trained in an unsupervised way as a language model on a large corpus and then fine-tuned on a (usually) small-sized corpus with manually labelled keywords. By conducting experiments on two different domains, computer science articles and news, we show that the language model pretraining allows the algorithm to successfully adapt to a specific domain and grasp more semantic information of the text, which drastically reduces the needed amount of labelled data for training the keyword detector.

The transfer learning technique (Peters et al., 2018a; Howard & Ruder, 2018), which has recently become a well established procedure in the field of natural language processing (NLP), in a large majority of cases relies on very large unlabelled textual resources used for language model pretraining. For example, a well known English BERT model (Devlin et al., 2018) was pretrained on the Google Books Corpus (Goldberg & Orwant, 2013) (800 million tokens) and Wikipedia (2,500 million tokens). On the other hand, we show that smaller unlabelled domain specific corpora (87 million tokens for computer science and 232 million tokens for news domain) can be successfully used for unsupervised pretraining, which makes the proposed approach easily transferable to languages with less textual resources and also makes training more feasible in terms of time and computer resources available.

Unlike most other proposed state-of-the-art neural keyword extractors (Meng et al., 2017, 2019; Yuan et al., 2019), we do not employ recurrent neural networks but instead opt for a transformer architecture

[1] Code is available under the MIT license at `https://github.com/EMBEDDIA/tnt_kid/`.

(Vaswani et al., 2017), which has not been widely employed for the task at hand. In fact, the study by Sahrawat et al. (2020) is the only study we are aware of that employs transformers for the keyword extraction task. Another difference between our approach and recent state-of-the-art approaches is the task formulation. While Meng et al. (2017, 2019) and Yuan et al. (2019) formulate a keyword extraction task as a sequence-to-sequence generation task, where the classifier is trained to generate an output sequence of keyword tokens step by step according to the input sequence and the previous generated output tokens, we formulate a keyword extraction task as a sequence labelling task, similar as in Gollapalli et al. (2017), Luan et al. (2017) and Sahrawat et al. (2020).

Besides presenting a novel keyword extraction procedure, the study also offers an extensive error analysis, in which the visualization of transformer attention heads is used to gain insights into inner workings of the model and in which we pinpoint key factors responsible for the differences in performance of TNT-KID and other state-of-the-art approaches. Finally, this study also offers a systematic evaluation of several building blocks and techniques used in a keyword extraction workflow in the form of an ablation study. Besides determining the extent to which transfer learning affects the performance of the keyword extractor, we also compare two different pretraining objectives, autoregressive language modelling and masked language modelling (Devlin et al., 2018), and measure the influence of transformer architecture adaptations, a choice of input encoding scheme and the addition of part-of-speech (POS) tags information on the performance of the model.

2.1 Related work

This section overviews selected methods for keyword extraction, supervised in Section 2.1.1 and unsupervised in Section 2.1.2. The related work is focused on the newest keyword extraction methods; therefore, for a more comprehensive survey of older methods, we refer the reader to Hasan & Ng (2014).

2.1.1 Supervised keyword extraction methods

Traditional supervised approaches to keyword extraction considered the task as a two step process (the same is true for unsupervised approaches). First, a number of syntactic and lexical features are used to extract keyword candidates from the text. Secondly, the extracted candidates are ranked according to different heuristics and the top n candidates are selected as keywords (Yuan et al., 2019). One of the first supervised approaches to keyword extraction was proposed by Witten et al. (2005), whose algorithm named KEA uses only TF-IDF and the term's position in the text as features for term identification. These features are fed to the Naive Bayes classifier, which is used to determine for each word or phrase in the text if it is a keyword or not. Medelyan et al. (2009) build on the KEA approach and proposed the *Maui* algorithm, which also relies on the Naive Bayes classifier for candidate selection but employs additional semantic features, such as e.g., *node degree*, which quantifies the semantic relatedness of a candidate to other candidates, and *Wikipedia-based keyphraseness*, which is the likelihood of a phrase being a link in the Wikipedia.

A more recent supervised approach is the so-called sequence labelling approach to keyword extraction by Gollapalli et al. (2017), where the idea is to train a keyword tagger using token-based linguistic, syntactic and structural features. The approach relies on a trained Conditional Random Field (CRF) tagger and the authors demonstrated that this approach is capable of working on-par with slightly older state-of-the-art systems that rely on information from the Wikipedia and citation networks, even if only within-document features are used. Another sequence labelling approach proposed by Luan et al. (2017) builds a sophisticated neural network by combing an input layer comprising a concatenation of word, character and part-of-speech embeddings, a bidirectional Long Short-Term Memory (BiLSTM) layer and a CRF tagging layer. They also propose a new semi-supervised graph based training regime for training the network.

Some of the most recent state-of-the-art approaches to keyword detection consider the problem as a sequence-to-sequence generation task. The first research leveraging this tactic was proposed by Meng et al. (2017), employing a generative model for keyword prediction with a recurrent encoder-decoder framework with an attention mechanism capable of detecting keywords in the input text sequence and also potentially finding keywords that do not appear in the text. Since finding absent keywords involves a very hard problem of finding a correct class in a set of usually thousands of unbalanced classes, their model also employs a copying mechanism (Gu et al., 2016) based on positional information, in order to allow the model to find important keywords present in the text, which is a much easier problem.

The model proposed by Meng et al. (2017) has been somewhat improved by investigating different ways in which the target keywords can be fed to a classifier during the training phase. While the original system used a so-called *one-to-one* approach, where a training example consists of an input text and a single keyword, the improved model (Meng et al., 2019) now employs a *one-to-seq* approach, where an input text is matched with a concatenated sequence made of all the keywords for a specific text. The study also shows that the order of the keywords in the text matters. The best performing model from Meng et al. (2019), named CopyRNN, is used in our experiments for the comparison with the state-of-the-art (see Section 2.3). A *one-to-seq* approach has been even further improved by Yuan et al. (2019), who incorporated two diversity mechanisms into the model. The mechanisms (called *semantic coverage* and *orthogonal regularization*) constrain the over-all inner representation of a generated keyword sequence to be semantically similar to the overall meaning of the source text and therefore force the model to produce diverse keywords. The resulting model leveraging these mechanisms has been named CatSeqD and is also used in our experiments for the comparison of TNT-KID with the state-of-the-art.

A further improvement of the generative approach for keyword detection has been proposed by Chan et al. (2019), who integrated a reinforcement learning (RL) objective into the keyphrase generation approach proposed by Yuan et al. (2019). This was done by introducing an adaptive reward function that encourages the model to generate sufficient amount of accurate keyphrases. They also proposed a new Wikipedia based evaluation method that can more robustly evaluate the quality of the predicted keyphrases by also considering name variations of the ground-truth keyphrases.

We are aware of one study that tackled keyword detection with transformers. Sahrawat et al. (2020) fed contextual embeddings generated using several transformer and recurrent architectures (BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), GPT-2 (Radford et al., 2019), ELMo (Peters et al., 2018b), etc.) into two distinct neural architectures, a bidirectional Long short-term memory network (BiLSTM) and a BiLSTM network with an additional Conditional random fields layer (BiLSTM-CRF). Same as in Gollapalli et al. (2017), they formulate a keyword extraction task as a sequence labelling approach, in which each word in the document is assigned one of the three possible labels: k_b denotes that the word is the first word in a keyphrase, k_i means that the word is inside a keyphrase, and k_o indicates that the word is not part of a keyphrase.

The study shows that contextual embeddings generated by transformer architectures generally perform better than static (e.g., FastText embeddings (Bojanowski et al., 2017)) and among them BERT showcases the best performance. Since all of the keyword detection experiments are conducted on scientific articles, they also test SciBERT (Beltagy et al., 2019), a version of BERT pretrained on a large multi-domain corpus of scientific publications containing 1.14M papers sampled from Semantic Scholar. They observe that this genre specific pretraining on texts of the same genre as the texts in the keyword datasets, slightly improves the performance of the model. They also report significant gains in performance when the BiLSTM-CRF architecture is used instead of BiLSTM.

The neural sequence-to-sequence models are capable of outperforming all older supervised and unsupervised models by a large margin but do require a very large training corpora with tens of thousands of documents for successful training. This means that their use is limited only to languages (and genres) in which large corpora with manually labelled keywords exist. On the other hand, the study by Sahrawat et al. (2020) indicates that the employment of contextual embeddings reduces the need for a large dataset with manually labelled keywords. These models can therefore be deployed directly to smaller datasets

by leveraging semantic information already encoded in contextual embeddings.

2.1.2 Unsupervised keyword extraction methods

The previous section discussed recently emerged methods for keyword extraction that operate in a supervised learning setting and can be data-intensive and time consuming. Unsupervised keyword detectors can tackle these two problems, yet at the cost of the reduced overall performance.

Unsupervised approaches need no training and can be applied directly without relying on a gold standard document collection. They can be divided into statistical and graph-based methods:

- Statistical methods, such as KP-MINER (El-Beltagy & Rafea, 2009), RAKE (Rose et al., 2010) and YAKE (Campos et al., 2018a,b), use statistical characteristics of the texts to capture keywords.
- Graph-based methods, such as TextRank (Mihalcea & Tarau, 2004), Single Rank (Wan & Xiao, 2008), TopicRank (Bougouin et al., 2013), Topical PageRank (Sterckx et al., 2015) and RaKUn (Škrlj et al., 2019) build graphs to rank words based on their position in the graph.

Among the statistical approaches, the state-of-the-art keyword extraction algorithm is YAKE (Campos et al., 2018a,b). It defines a set of features capturing keyword characteristics which are heuristically combined to assign a single score to every keyword. These features include casing, position, frequency, relatedness to context and dispersion of a specific term.

One of the first graph-based methods for keyword detection is TextRank (Mihalcea & Tarau, 2004), which first extracts a lexical graph from text documents and then leverages Google's PageRank algorithm to rank vertices in the graph according to their importance inside a graph. This approach was somewhat upgraded by TopicRank (Bougouin et al., 2013), where candidate keywords are additionally clustered into topics and used as vertices in the graph. Keywords are detected by selecting a candidate from each of the top-ranked topics. The most recent graph-based keyword detector is RaKUn (Škrlj et al., 2019) that employs several new techniques for graph construction and vertice ranking. First, initial lexical graph is expanded and adapted with the introduction of meta-vertices, i.e. aggregates of existing vertices. Second, for keyword detection and ranking, a graph-theoretic *load centrality* measure is used along with the implemented graph redundancy filters.

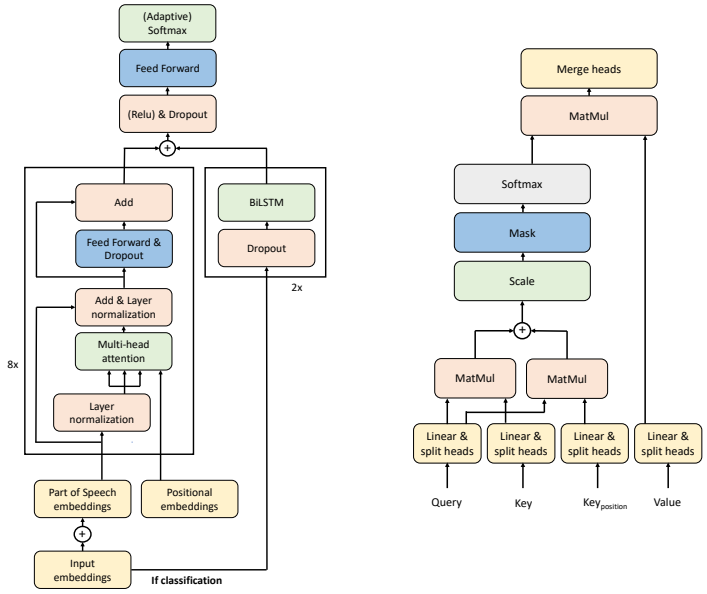
2.2 Methodology

This section presents the methodology of our approach. Section 2.2.1 presents the architecture of the neural model, Section 2.2.2 covers the transfer learning techniques used, Section 2.2.3 explains how the final fine-tuning phase of the keyword detection workflow is conducted and Section 2.2.4 covers evaluation of the model.

2.2.1 Architecture

The model follows an architectural design of an original transformer encoder (Vaswani et al., 2017) and is presented in Figure 1a. In the GPT-2 architecture (Radford et al., 2019), the encoder consists of a normalization layer that is followed by a multi-head attention mechanism. A residual connection is employed around the attention mechanism, which is followed by another layer normalization. This is followed by the fully connected feed-forward and dropout layers, around which another residual connection is employed.

For two distinct training phases, language model pretraining and fine-tuning, two distinct "heads" are added on top of the encoder, which is identical for both phases and therefore allows for the transfer of weights from the pretraining phase to the fine-tuning phase. The language model head predicts the probability for each word in the vocabulary that it appears at a specific position in the sequence and



(a) Model architecture. **(b)** The attention mechanism.

Figure 1: TNT-KID's architecture overview.

consists of a dropout layer and a feed forward layer of size $\mathrm{SL} * |V|$, where SL stands for sequence length (i.e., a number of words in the input text) and $|V|$ stands for the vocabulary size. This is followed by the adaptive softmax layer (Grave et al., 2017) (see description below).

During fine-tuning, the language model head is replaced with a token classification head, in which we apply ReLu non-linearity and dropout to the encoder output, and then feed the output to the feed forward classification layer of size $\mathrm{SL} * \mathrm{NC}$, where NC stands for the number of classes (in our case 2, since we model keyword extraction as a binary classification task). Finally, a softmax layer is added in order to obtain probabilities for each class.

We also propose some significant modifications of the original GPT-2 architecture described above:

1. Re-parametrization of the attention mechanism
2. Addition of the part-of-speech (POS) tag sequence input
3. Replacement of the standard input embedding layer and softmax function with adaptive input representations (Baevski & Auli, 2018) and an adaptive softmax (Grave et al., 2017)
4. Addition of the two bidirectional Long short-term memory (BiLSTM) layers to the output of the transformer encoder
5. An employment of the BiLSTM-CRF classification head on top of the transformer encoder (Sahrawat et al., 2020)

The re-parametrization of the attention mechanism (1) allows to model the relation between a token and its position more directly (see Figure 1b). Note that standard scaled dot-product attention (Vaswani et al., 2017) requires three inputs, a so-called *query, key, value* matrix representations of the embedded input sequence and its positional information (i.e., element wise addition of input embeddings and positional embeddings) and the idea is to obtain attention scores (in a shape of an attention matrix) for each relation between tokens inside these inputs by first multiplying *query* (Q) and transposed *key* (K) matrix representations, applying scaling and softmax functions, and finally multiplying the resulting normalized matrix with the *value* (V) matrix, or more formally,

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where d_k represents the scaling factor, usually corresponding to the first dimension of the *key* matrix. On the other hand, we propose to add an additional positional input representation matrix K_{position} and model attention with the following equation:

$$\text{Attention}(Q, K, V, K_{pos}) = \text{softmax}\left(\frac{QK^T + QK^T_{position}}{\sqrt{d_k}}\right)V$$

The reason behind this modification is connected with the hypothesis, that token position is important in the keyword identification task and with this re-parametrization the model is capable of directly modelling the importance of relation between each token and each position. Note that we use relative positional embeddings for representing the positional information, same as in Dai et al. (2019), where the main idea is to only encode the relative positional information in the hidden states instead of the absolute positional information.

Second, besides the text input, we also experiment with the additional part-of-speech (POS) tag sequence as an input (2). This sequence is first embedded and then added to the word embedding matrix. Note that this additional input is optional and is not included in the model for which the results are presented in Section 2.3.3 due to marginal effect on the performance of the model in the proposed experimental setting (see Section 2.5).

While the modifications presented above affect both training phases (i.e., the language model pretraining and the token classification fine-tuning), the third modification only affects the language model pretraining (see Section 2.2.2) and involves replacing the standard input embedding layer and softmax function with adaptive input representations (Baevski & Auli, 2018) and an adaptive softmax (Grave et al., 2017) (3). The main idea is to exploit the unbalanced word distribution to form word clusters containing words with similar appearance probabilities. The entire vocabulary is split into a smaller cluster containing about 10 percent of words that appear most frequently, a second slightly bigger cluster that contains words that appear less frequently and a third cluster that contains all the other words that appear rarely in the corpus. During language model training, instead of predicting an entire vocabulary distribution at each time step, the model first tries to predict a cluster in which a target word appears in and after that predicts a vocabulary distribution just for the words in that cluster. Since in a large majority of cases the target word belongs to the smallest cluster containing most frequent words, the model in most cases only needs to generate probability distribution for a tenth of a vocabulary, which drastically reduces the memory requirements and time complexity of the model at the expense of a marginal drop in performance.

We also present the modification, which only affects the fine-tuning token classification phase (see Section 2.2.3). During this phase, a two layer randomly initialised encoder, consisting of dropout and two bidirectional Long short-term memory (BiLSTM) layers, is added (with element-wise summation) to the output of the transformer encoder (4). The initial motivation behind this adaptation is connected with findings from the related work which suggest that recurrent layers are quite successful at modelling positional importance of tokens in the keyword detection task (Meng et al., 2017; Yuan et al., 2019) and by the study of Sahrawat et al. (2020), who also reported good results when a BiLSTM classifier and contextual embeddings generated by transformer architectures were employed for keyword detection. Also, the results of the initial experiments suggested that some performance gains can in fact be achieved by employing this modification.

In terms of computational complexity, a self-attention layer complexity is $O(n^2 * d)$ and the complexity of the recurrent layer is $O(n * d^2)$, where n is the sequence length and d is the embedding size (Vaswani et al., 2017). This means that the complexity of the transformer model with an additional BiLSTM encoder is therefore $O(n^2 * d^2)$. When it comes to the number of parameters, the standard TNT-KID model employs sequence size of 256, embedding size of 512 and 8 attention layers, resulting in altogether

$256^2 * 512 * 8 = 268435456$ parameter. By adding the recurrent encoder with two recurrent bidirectional layers (which is the same as adding 4 recurrent layers, since each bidirectional layer contains two unidirectional LSTM layers), the number of parameters increases by $256 * 512^2 * 4 = 268435456$. In practice this means that the model with the additional recurrent encoder conducts token classification roughly two times slower than the model without the encoder. Note that this addition does not affect the language model pretraining, which tends to be the more time demanding task due to larger corpora involved.

Additionally, we also experiment with an employment of the BiLSTM-CRF classification head on top of the transformer encoder (5), in order to compare our proposed approach to the approach proposed by Sahrawat et al. (2020) (see Section 2.5 for more details about the results of this experiment). For this experiment, during the fine-tuning token classification phase, the token classification head described above is replaced with a BiLSTM-CRF classification head proposed by Sahrawat et al. (2020), containing one BiLSTM layer and a CRF (Lafferty et al., 2001) layer.[2] Outputs of the BiLSTM $f = f_1, ..., f_n$ are fed as inputs to a CRF layer, which returns the output score $s(f, y)$ for each possible label sequence according to the following equation:

$$s(f, y) = \sum_{t=1}^{n} \tau_{y_{t-1}, y_t} + f_{t, y_t}$$

τ_{y_{t-1}, y_t} is a transition matrix representing the transition score from class y_{t-1} to y_t. The final probability of each label sequence score is generated by exponentiating the scores and normalizing over all possible output label sequences:

$$p(y|f) = \frac{exp(s(f, y))}{\sum_{y'} exp(s(f', y'))}$$

To find the optimal sequence of labels efficiently, the CRF layer uses the Viterbi algorithm (Forney, 1973).

2.2.2 Transfer learning

Our approach relies on a transfer learning technique (Howard & Ruder, 2018; Devlin et al., 2018), where a neural model is first pretrained as a language model on a large corpus. This model is then fine-tuned for each specific keyword detection task on each specific manually labelled corpus by adding and training the token classification head described in the previous section. With this approach, the syntactic and semantic knowledge of the pretrained language model is transferred and leveraged in the keyword detection task, improving the detection on datasets that are too small for the successful semantic and syntactic generalization of the neural model.

In the transfer learning scenario, two distinct pretraining objectives can be considered. First is the autoregressive language modelling where the task can be formally defined as predicting a probability distribution of words from the fixed size vocabulary V, for word w_t, given the historical sequence $w_{1:t-1} = [w_1, ..., w_{t-1}]$. This pretraining regime was used in the GPT-2 model (Radford et al., 2019) that we modified. Since in the standard transformer architecture self-attention is applied to an entire surrounding context of a specific word (i.e. the words that appear after a specific word in each input sequence are also used in the self-attention calculation), we employ obfuscation masking to the right context of each word when the autoregressive language model objective is used, in order to restrict the information only to the prior words in the sentence (plus the word itself) and prevent target leakage (see Radford et al. (2019) for details on the masking procedure).

[2] Note that in the experiments in which we employ BiLSTM-CRF, we do not add an additional two layer BiLSTM encoder described above to the output of the transformer encoder.

Another option is a masked language modelling objective, first proposed by Devlin et al. (2018). Here, a percentage of words from the input sequence is masked in advance, and the objective is to predict these masked words from an unmasked context. This allows the model to leverage both left and right context, or more formally, the token w_t is also determined by sequence of tokens $w_{t+1:n} = [w_{t+1}, ..., w_{t+n}]$. We follow the masking procedure described in the original paper by Devlin et al. (2018), where 15 percent of words are randomly designated as targets for prediction, out of which 80 percent are replaced by a masked token ($< mask >$), 10 percent are replaced by a random word and 10 percent remain intact.

The final output of the model is a softmax probability distribution calculated over the entire vocabulary, containing the predicted probabilities of appearance (P) for each word given its left (and in case of the *masked language modelling objective* also right) context. Training therefore consists of the minimization of the negative log-loss (NLL) on the batches of training corpus word sequences by backpropagation through time:

$$\text{NLL} = -\sum_{i=1}^{n} \log P(w_i | w_{1:i-1}) \tag{1}$$

While the *masked language modelling* objective might outperform autoregressive language modelling objective in a setting where a large pretraining corpus is available (Devlin et al., 2018) due to the inclusion of the right context, these two training objectives have at least to our knowledge never been compared in a setting where only a relatively small domain specific corpus is available for the pretraining phase. For more details about the performance comparison of these two pretraining objectives, see Section 2.5.

2.2.3 Keyword identification

Since each word in the sequence can either be a keyword (or at least part of the keyphrase) or not, the keyword tagging task can be modeled as a binary classification task, where the model is trained to predict if a word in the sequence is a keyword or not.[3] Figure 2 shows an example of how an input text is first transformed into a numerical sequence that is used as an input of the model, which is then trained to produce a sequence of zeroes and ones, where the positions of ones indicate the positions of keywords in the input text.

Since a large majority of words in the sequence are not keywords, the usage of a standard NLL function (see Equation 1), which would simply calculate a sum of log probabilities that a word is either a keyword or not for every input word sequence, would badly affect the recall of the model since the majority negative class would prevail. To solve this problem and maximize the recall of the system, we propose a custom classification loss function, where probabilities for each word in the sequence are first aggregated into two distinct sets, one for each class. For example, text "*The advantage of this is to include distributed interactions between the UDDI clients.*" in Figure 2 would be split into two sets, first one containing probabilities for all the words in the input example which are not keywords (*The, advantage, of, this, is, to, include, between, the, clients, .*), and the other containing probabilities for all the words in the input example that are keywords or part of keyphrases (*distributed, interactions, UDDI*). Two NLLs are calculated, one for each probability set, and both are normalized with the size of the set. Finally, the NLLs are summed. More formally, the loss is computed as follows. Let $W = \{w_i\}_{i=1}^{n}$ represent an enumerated sequence of tokens for which predictions are obtained. Let p_i represent the predicted probabilities for the i-th token that it either belongs or does not belong to the ground truth class. The o_i represents the output weight vector of the neural network for token i and j corresponds to the number of classes (two in our case as the word can be a keyword or not). Predictions are in this work obtained via a log-softmax transform (lso), defined as follows (for the i-th token):

[3] Note that this differs from the sequence labelling approach proposed by Sahrawat et al. (2020), where each word in the document is assigned one of three possible labels (see Section 2.1 for details).

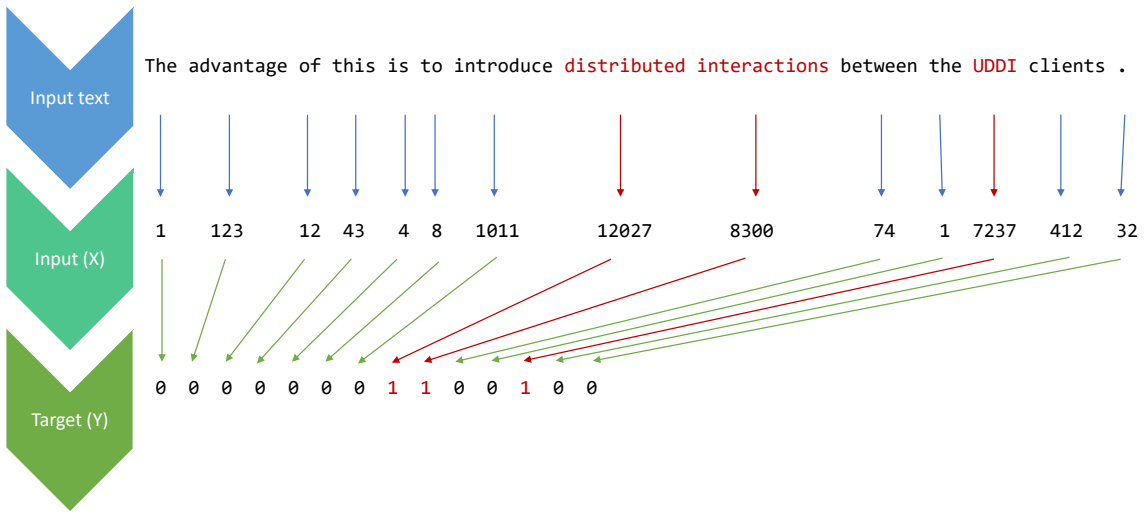


Figure 2: Encoding of the input text "*The advantage of this is to introduce distributed interactions between the UDDI clients.*" with keywords *distributed interactions* and *UDDI*. In the first step, the text is converted into a numerical sequence, which is used as an input to the model. The model is trained to convert this numerical sequence into a sequence of zeroes and ones, where the ones indicate the position of a keyword.

$$p_i = \text{lso}(o_i) = \log \frac{\exp(o_i)}{\sum_j \exp(o_j)}.$$

The loss function is comprised from two main parts. Let $K_+ \subseteq W$ represent tokens that are keywords and $K_- \subseteq W$ the set of tokens that are **not** keywords. Note that $|K_- \cup K_+| = n$, i.e., the two sets cover all considered tokens for which predictions are obtained. During loss computation, only the probabilities of the ground truth class are considered. We mark them with p_i^+ or p_i^-. Then the final loss is computed as

$$\text{Loss} = L_+ + L_- = -\frac{1}{|K_+|} \sum_{w_i \in K_+} p_i^+ \quad + (-\frac{1}{|K_-|} \sum_{w_i \in K_-} p_i^-)$$

Note that even though all predictions are given as an argument, the two parts of the loss address different token indices (i).

In order to produce final set of keywords for each document, tagged words are extracted from the text and duplicates are removed. Note that a sequence of ones is always interpreted as a multi-word keyphrase and not as a combination of one-worded keywords (e.g., *distributed interactions* from Figure 2 is considered as a single multi-word keyphrase and not as two distinct one word keywords). After that, the following filtering is conducted:

- If a keyphrase is longer than four words, it is discarded.
- Keywords containing punctuation (with the exception of dashes and apostrophes) are removed.

- The detected keyphrases are ranked and arranged according to the softmax probability assigned by the model in a descending order.

2.2.4 Evaluation

To asses the performance of the model, we measure $F_1@k$ score, a harmonic mean between Precision@k and Recall@k. In a ranking task, we are interested in precision at rank k. This means that only the keywords ranked equal to or better than k are considered and the rest are disregarded. Precision is the ratio of the number of correct keywords returned by the system divided by the number of all keywords returned by the system, or more formally:

$$precision = \frac{|correct \quad returned \quad keywords@k|}{|returned \quad keywords|}$$

Recall@k is the ratio of the number of correct keywords returned by the system and ranked equal to or better than k divided by the number of correct ground truth keywords:

$$recall = \frac{|correct \quad returned \quad keywords@k|}{|correct \quad keywords|}$$

Due to the high variance of a number of ground truth keywords, this type of recall becomes problematic if k is smaller than the number of ground truth keywords, since it becomes impossible for the system to achieve a perfect recall. (Similar can happen to precision@k, if the number of keywords in a gold standard is lower than k, and returned number of keywords is fixed at k.)

Finally, we formally define $F_1@k$ as a harmonic mean between Precision@k and Recall@k:

$$F_1@k = 2 * \frac{P@k * R@k}{P@k + R@k}$$

In order to compare the results of our approach to other state-of-the-art approaches, we use the same evaluation methodology as Yuan et al. (2019) and Meng et al. (2019), and measure F1@k with k being either 5 or 10. Note that F1@k is calculated as a harmonic mean of macro-averaged precision and recall, meaning that precision and recall scores for each document are averaged and the F1 score is calculated from these averages. Same as in the related work, lowercasing and stemming are performed on both the gold standard and the generated keywords (keyphrases) during the evaluation. Only keywords that appear in the text of the documents (present keywords)[4] were used as a gold standard and the documents containing no present keywords were removed, in order to make the results of the conducted experiments comparable with the reported results from the related work.

2.3 Experiments

We first present the datasets used in the experiments. This is followed by the experimental design and the results achieved by TNT-KID in comparison to the state-of-the-art.

2.3.1 Keyword extraction datasets

Experiments were conducted on seven datasets from two distinct genres, scientific papers about computer science and news. The following datasets from the computer science domain are used:

[4]Note that scientific and news articles often list keywords that do not appear in the text of the article. For example, an NLP paper would often list "*Text mining*" as a keyword of the paper, even though the actual phrase does not appear in the text of the paper.

- **KP20k (Meng et al., 2017)**: This dataset contains titles, abstracts, and keyphrases of 570,000 scientific articles from the field of computer science. The dataset is split into train set (530,000), validation set (20,000) and test set (20,000).
- **Inspec (Hulth, 2003)**: The dataset contains 2,000 abstracts of scientific journal papers in computer science collected between 1998 and 2002. Two sets of keywords are assigned to each document, the controlled keywords that appear in the Inspec thesaurus, and the uncontrolled keywords, which are assigned by the editors. Only uncontrolled keywords are used in the evaluation, same as by Meng et al. (2017), and the dataset is split into 500 test papers and 1500 train papers.
- **Krapivin (Krapivin et al., 2009)**: This dataset contains 2,304 full scientific papers from computer science domain published by ACM between 2003 and 2005 with author-assigned keyphrases. 460 papers from the dataset are used as a test set and the others are used for training. Only titles and abstracts are used in our experiments.
- **NUS (Nguyen & Kan, 2007)**: The dataset contains titles and abstracts of 211 scientific conference papers from the computer science domain and contains a set of keywords assigned by student volunters and a set of author assigned keywords, which are both used in evaluation.
- **SemEval (Kim et al., 2010)**: The dataset used in the SemEval-2010 Task 5, Automatic Keyphrase Extraction from Scientific Articles, contains 244 articles from the computer science domain collected from the ACM Digital Library. 100 articles are used for testing and the rest are used for training. Again, only titles and abstracts are used in our experiments, the article's content was discarded.

From the news domain, three datasets with manually labelled gold standard keywords are used:

- **KPTimes (Gallina et al., 2019)**: The corpus contains 279,923 news articles containing editor assigned keywords that were collected by crawling New York Times news website[5]. After that, the dataset was randomly divided into training (92.8 percent), development (3.6 percent) and test (3.6 percent) sets.
- **JPTimes (Gallina et al., 2019)**: Similar as **KPTimes**, the corpus was collected by crawling Japan Times online news portal[6]. The corpus only contains 10,000 English news articles and is used in our experiments as a test set for the classifiers trained on the **KPTimes** dataset.
- **DUC (Wan & Xiao, 2008)**: The dataset consists of 308 English news articles and contains 2,488 hand labelled keyphrases.

The statistics about the datasets that are used for training and testing of our models are presented in Table 1. Note that there is a big variation in dataset sizes in terms of number of documents (column *No. docs*), and in an average number of keywords (column *Avg. kw.*) and present keywords per document (columns *Avg. present kw.*), ranging from 2.35 present keywords per document in *KPTimes-valid* to 7.79 in *DUC-test*.

2.3.2 Experimental design

We conducted experiments on the datasets described in Section 2.3.1. First, we lowercased and tokenized all datasets. We experimented with two tokenization schemes, word tokenization and Sentencepiece (Kudo & Richardson, 2018) byte-pair encoding (see Section 2.5 for more details on how these two tokenization schemes affect the overall performance). During both tokenization schemes, a special $< eos >$ token is used to indicate the end of each sentence. For the best performing model, for which the results are presented in Section 2.3.3, byte-pair encoding was used.[7] For generating the additional POS

[5] `https://www.nytimes.com`

[6] `https://www.japantimes.co.jp`

[7] As in byte-pair encoding the subword information is already considered, there is no need to perform additional prepossessing techniques, such as lemmatization or stemming.

Table 1: Datasets used for empirical evaluation of keyword extraction algorithms. *No.docs* stands for number of documents, *Avg. doc. length* stands for average document length in the corpus, *Avg. kw.* stands for average number of keywords per document in the corpus, *% present kw.* stands for the percentage of keywords that appear in the corpus (i.e., percentage of document's keywords that appear in the text of the document) and *Avg. present kw.* stands for the average number of keywords per document that actually appear in the text of the specific document.

Dataset	No. docs	Avg. doc. length	Avg. kw.	% present kw.	Avg. present kw.
Computer science papers					
KP20k-train	530,000	156.34	5.27	62.43	3.29
KP20k-valid	20,000	156.55	5.26	62.30	3.28
KP20k-test	20,000	156.52	5.26	62.55	3.29
Inspec-valid	1500	125.21	9.57	76.92	7.36
Inspec-test	500	121.82	9.83	78.14	7.68
Krapivin-valid	1844	156.65	5.24	54.34	2.85
Krapivin-test	460	157.76	5.74	55.66	3.20
NUS-test	211	164.80	11.66	50.47	5.89
SemEval-valid	144	166.86	15.67	45.43	7.12
SemEval-test	100	183.71	15.07	44.53	6.71
News articles					
KPTimes-train	259,923	783.32	5.03	47.30	2.38
KPTimes-valid	10,000	784.65	5.02	46.78	2.35
KPTimes-test	10,000	783.47	5.04	47.59	2.40
JPTimes-test	10,000	503.00	5.03	76.73	3.86
DUC-test	308	683.14	8.06	96.62	7.79

tag sequence input described in Section 2.2.1, which was **not** used in the best performing model, Averaged Perceptron Tagger from the NLTK library (Loper & Bird, 2002) was used. The neural architecture was implemented in PyTorch (Paszke et al., 2019).

In the pretraining phase, two language models were trained for up to ten epochs, one on the concatenation of all the texts from the computer science domain and the other on the concatenation of all the texts from the news domain. Overall, the language model train set for computer science domain contained around 87 million tokens and the news train set about 232 million tokens. These small sizes of the language model train sets enable relatively fast training and smaller model sizes (in terms of number of parameters) due to the reduced vocabulary.

After the pretraining phase, the trained language models were fine-tuned on each dataset's *validation* sets, i.e., datasets with a suffix *-valid* (see Table 1, which were randomly split into 80 percent of documents used for training and 20 percent of documents used for validation. The documents containing more than 256 tokens are truncated, while the documents containing less than 256 tokens are padded with a special $< pad >$ token at the end. Each model was fine-tuned for a maximum of 10 epochs and after each epoch the trained model was tested on the documents chosen for validation. The model that showed the best performance on this set of validation documents (in terms of F@10 score) was used for keyword detection on the test set. Validation sets were also used to determine the best hyperparameters of the model and all combinations of the following hyperparameter values were tested before choosing the best combination, which is written in bold in the list below and on average worked best for all the datasets in both domains[8]:

- Learning rates: 0.00005, 0.0001, **0.0003**, 0.0005, 0.001
- Embedding size: 256, **512**
- Number of attention heads: 4, **8**, 12
- Sequence length: 128, **256**

[8]Note that the same set of hyperparameters are also used in the pretraining phase.

- Number of attention layers: 4, **8**, 12

Note that in our experiments, we use the same splits as in related work (Meng et al., 2019, 2017; Gallina et al., 2019) for all datasets with predefined splits i.e., all datasets with a validation set (see Table 1. The exceptions are NUS, DUC and JPTimes datasets with no available predefined validation-test splits. For NUS and DUC, 10-fold cross-validation is used and the model used for keyword detection on the JPTimes-test dataset was fine-tuned on the KPTimes-valid dataset. Another thing to consider is that in the related work by Yuan et al. (2019), Meng et al. (2017) and Gallina et al. (2019), to which we are comparing, large datasets KPTimes-train and KP20k-train with 530,000 documents and 260,00 documents, respectively, are used for the classification model training and these trained models are applied on all test sets from the matching domain. On the other hand, we do not train our classification models on these two large train sets but instead use smaller KPTimes-valid and KP20k-valid datasets for training, since we argue that, due to language model pretraining, fine-tuning the model on a relatively small labelled dataset is sufficient for the model to achieve competitive performance. We do however conduct the language model pretraining on the concatenation of all the texts from the computer science domain and the news domain as explained above, and these two corpora also contain texts from KPTimes-train and KP20k-train datasets.

2.3.3 Keyword extraction results and comparison to the state-of-the-art

In Table 2, we present the results achieved by TNT-KID and a number of algorithms from the related work on the datasets presented in Table 1. Evaluation measures were presented in Section 2.2.4. Only keywords which appear in a text (present keywords) were used as a gold standard in order to make the results of the conducted experiments comparable with reported results from the related work. Note that TF-IDF, TextRank, YAKE and RaKUn algorithms are unsupervised and do not require any training, KEA, Maui, GPT-2, GPT-2 + BiLSTM-CRF and TNT-KID were trained on the different *validation* set for each of the datasets, and CopyRNN and CatSeqD were trained on the large KP20k-train dataset for keyword detection on computer science domain, and on the KPTimes-train dataset for keyword detection on the news domain, since they require a large train set for competitive performance.

For RaKUn (Škrlj et al., 2019) and YAKE (Campos et al., 2020), we report results for default hyperparameter settings, since the authors of RaKUn and YAKE claim that a single hyperparameter set can offer sufficient performance across multiple datasets. We used the author's official github implementations[9] in the experiments. For KEA and Maui we do not conduct additional testing on corpora for which results are not available in the related work (KPTimes, JPTimes and DUC corpus) due to bad performance of the algorithms on all the corpora for which results are available. Finally, for TF-IDF and TextRank we report results from the related work where available (Yuan et al., 2019) and use the implementation of the algorithms from the Python Keyphrase Extraction (PKE) library[10] to generate unavailable results. Same as for RaKUn and YAKE, default hyperparameters are used.

For KEA, Maui, CopyRNN and CatSeqD, we report results for the computer science domain published in Yuan et al. (2019) and for the news domain we report results for CopyRNN published in Gallina et al. (2019). The results that were not reported in the related work are results for CatSeqD on KPTimes, JPTimes and DUC, since this model was originally not tested on these three datasets, and the F1@5 score results for CopyRNN on KPTimes and JPTimes. Again, author's official github implementations[11] were used for training and testing of both models. The models were trained and tested on the large KPTimes-train dataset with a help of a script supplied by the authors of the papers. Same hyperparameters that were used for KP20k training in the original papers (Yuan et al., 2019; Meng et al., 2019) were used.

We also report results for the unmodified pretrained GPT-2 (Radford et al., 2019) model with a standard feed forward token classification head, and a pretrained GPT-2 with a BiLSTM-CRF token classification

[9] `https://github.com/SkBlaz/rakun` and `https://github.com/LIAAD/yake`
[10] `https://github.com/boudinfl/pke`
[11] `https://github.com/memray/OpenNMT-kpg-release`

Table 2: Empirical evaluation of state-of-the-art keyword extractors. Results marked with * were obtained by our implementation or reimplementation of the algorithm and results without * were reported in the related work.

	Unsupervised approaches				Supervised approaches						
	TF-IDF	TextRank	YAKE	RaKUn	KEA	Maui	CopyRNN	CatSeqD	GPT-2	GPT-2 + BiLSTM-CRF	TNT-KID
KP20k											
F1@5	0.072	0.181	0.141*	0.177*	0.046	0.005	0.317	**0.348**	0.252*	0.339*	0.342*
F1@10	0.094	0.151	0.146*	0.160*	0.044	0.005	0.273	0.298	0.256*	0.342*	**0.346***
Inspec											
F1@5	0.160	0.286	0.204*	0.101*	0.022	0.035	0.244	0.276	0.293*	**0.467***	0.447*
F1@10	0.244	0.339	0.223*	0.108*	0.022	0.046	0.289	0.333	0.335*	**0.525***	**0.525***
Krapivin											
F1@5	0.067	0.185	0.215*	0.127*	0.018	0.005	0.305	**0.325**	0.210*	0.280*	0.301*
F1@10	0.093	0.160	0.196*	0.106*	0.017	0.007	0.266	0.285	0.214*	0.283*	**0.307***
NUS											
F1@5	0.112	0.230	0.159*	0.224*	0.073	0.004	0.376	**0.374**	0.274*	0.311*	0.350*
F1@10	0.140	0.216	0.196*	0.193*	0.071	0.006	0.352	0.366	0.305*	0.332*	**0.369***
SemEval											
F1@5	0.088	0.217	0.151*	0.167*	0.068	0.011	0.318	**0.327**	0.261*	0.214	0.291*
F1@10	0.147	0.226	0.212*	0.159*	0.065	0.014	0.318	0.352	0.295*	0.232	**0.355***
KPTimes											
F1@5	0.179*	0.022*	0.105*	0.168*	*	*	0.406*	0.424*	0.353*	0.439*	**0.469***
F1@10	0.151*	0.030*	0.118*	0.139*	*	*	0.393	0.424*	0.354*	0.440*	**0.469***
JPTimes											
F1@5	0.266*	0.012*	0.109*	0.225*	*	*	0.256*	0.238*	0.258*	**0.344***	0.337*
F1@10	0.229*	0.026*	0.135*	0.185*	*	*	0.246	0.238*	0.267*	0.346*	**0.360***
DUC											
F1@5	0.098*	0.120*	0.106*	0.189*	*	*	0.083	0.063*	0.247*	0.281*	**0.312***
F1@10	0.120*	0.181*	0.132*	0.172*	*	*	0.105	0.063*	0.277*	0.321*	**0.355***
Average											
F1@5	0.130	0.157	0.149	0.172	*	*	0.288	0.297	0.269*	0.334*	**0.356***
F1@10	0.152	0.166	0.170	0.153	*	*	0.280	0.295	0.288*	0.353*	**0.386***

head, as proposed in Sahrawat et al. (2020) and described in Section 2.2.1[12]. For these two models, we apply the same fine-tuning regime as for TNT-KID, i.e. we fine-tune the models for up to 10 epoch on each dataset's *validation* sets (see Table 1), which were randomly split into 80 percent of documents used for training and 20 percent of documents used for validation. The model that showed the best performance on this set of validation documents (in terms of F@10 score) was used for keyword detection on the test set. We use the default hyperparameters for both models and the original GPT-2 tokenization regime.

Overall, supervised neural network approaches drastically outperform all other approaches. Among them, TNT-KID performs the best on all eight datasets in terms of F1@10 but is outperformed by CatSeqD (on four datasets) or GPT-2+ BiLSTM-CRF (on two datasets) on six out of eight datasets in terms of F1@5. In terms of F1@10, CatSeqD performs competitively on KP20k, Krapivin, NUS, SemEval and KPTimes datasets but is outperformed by a large margin on three other datasets by both GPT-2 + BiLSTM-CRF and TNT-KID. To be more specific, in terms of F1@10, TNT-KID outperforms the CatSeqD approach by almost 20 percentage points on the Inspec dataset, on the DUC dataset, it outperforms CatSeqD by about 25 percentage points, and on JPTimes it outperforms CatSeqD by about 12 percentage points.

While the results of CopyRNN are in a large majority of cases very consistent with CatSeqD (CopyRNN performs slightly better than CatSeqD on DUC and JPTimes, and slightly worse on the other six datasets), results of TNT-KID are very similar to the results of GPT-2 + BiLSTM-CRF. In the majority of cases TNT-KID outperforms GPT-2 + BiLSTM-CRF by a small margin according to both criteria, the exceptions being Inspec and JPTimes, where GPT-2 + BiLSTM-CRF performs the best out of all approaches

[12]We use the implementation of GPT-2 from the Transformers library (`https://github.com/huggingface/transformers`) and use the Pytorch-crf library (`https://pytorch-crf.readthedocs.io/en/stable/`) for the implementation of the BiLSTM-CRF token classification head.

according to F1@5. Another exception is the SemEval dataset, where the GPT-2 + BiLSTM-CRF is outperformed by TNT-KID by a large margin of about 12 percentage points. On the other hand, a GPT-2 model with a standard token classification head does not perform competitively on most datasets.

When it comes to the F1@5 measure, TNT-KID performs competitively on all the datasets. It outperforms all other algorithms on two datasets (KPTimes and DUC) and on average still performs the best out of all algorithms (see row *average*). Nevertheless, the performance in terms of F1@5 is still noticeably worse than in terms of F1@10. The difference between TNT-KID and CatSeqD, which performs the best on four out of eight datasets in terms of F1@5, can be partially explained by the difference in training regimes and the fact that our system was designed to maximize recall (see Section 2.2). Since our system generally detects more keywords than CatSeqD and CopyRNN, it tends to achieve better recall, which offers a better performance when up to 10 keywords need to be predicted. On the other hand, a more conservative system that generally predicts less keywords tends to achieve a better precision, which positively affects the F1 score in a setting where only up to 5 keywords need to be predicted. This phenomenon will be analysed in more detail in Section 2.4, where we also discuss the very low results achieved by CopyRNN and CatSeqD on the DUC dataset.

When it comes to two other supervised approaches, KEA and Maui, they perform badly on all datasets they have been tested on and are outperformed by a large margin even by all unsupervised approaches. When we compare just unsupervised approaches, TextRank achieves by far the best results according to both measures on the Inspec dataset. This is the dataset with the on average shortest documents. On the other hand, TextRank performs uncompetitively in comparison to other unsupervised approaches on two datasets with much longer documents, KPTimes and JPTimes, where RaKUn and TF-IDF are the best unsupervised approaches, respectively. Interestingly, it achieves the highest F@10 score out of all unsupervised keyword detectors on the DUC dataset, which also contains long documents. Perhaps this could be explained by the average number of present keywords, which is much higher for DUC-test (7.79) than for KPTimes-test (2.4) and JPTimes-test (3.86) datasets.

Overall (see row *average*), TNT-KID offers the most robust performance on the test datasets and is closely followed by GPT-2 + BiLSTM. CopyRNN and CatSeqD are very close to each other according to both criteria. Out of unsupervised approaches, on average all of them offer surprisingly similar performance. According to the F@10 score, YAKE on average works slightly better than the second ranked TextRank and also in general offers more steady performance, since it gives the most consistent results on a variety of different datasets. Similar could be said for RaKUn, the best ranked unsupervised algorithm according to the F@5 score.

2.4 Error analysis

In this section we first analyse the reasons why transformer based TNT-KID is capable of outperforming other state-of-the-art neural keyword detectors, which employ a generative model, by a large margin on some of the datasets. Secondly, we gather some insights into the inner workings of the TNT-KID by a visual analysis of the attention mechanism.

2.4.1 Comparison between TNT-KID and CatSeqD

As was observed in Section 2.3.3, transformer based TNT-KID and GPT-2 + BiLSTM-CRF outperform generative models CatSeqD and CopyRNN by a large margin on the Inspec, JPTimes and DUC datasets. Here, we try to explain this discrepancy by focusing on the difference in performance between the best transformer based model, TNT-KID, and the best generative model, CatSeqD. The first hypothesis is connected with the statistical properties of the datasets used for training and testing, or more specifically, with the average number of keywords per document for each dataset. Note that CatSeqD is trained on the KP20k-train, when employed on the computer science domain, and on the KPTimes-train dataset, when employed on news. Table 1 shows that both of these datasets do not contain many present keywords per document (KP20k-train 3.28 and KPTimes-train 2.38), therefore training the model on

these datasets conditions it to be conservative in its predictions and to assign less keywords to each document than a more liberal TNT-KID. This gives the TNT-KID a competitive advantage on the datasets with more present keywords per document.

Figure 3 shows a correlation between the average number of present keywords per document for each dataset and the difference in performance in terms of F@10, measured as a difference between an F@10 score achieved by TNT-KID and an F@10 score achieved by CatSeqD. The difference in performance is the biggest for the DUC dataset (about 30 percentage points) that on average has the most keywords per document, 7.79, and second biggest for Inspec, in which an average document has 7.68 present keywords.

The above hypothesis explains why CatSeqD offers competitive performance on the KP20k-test, Krapivin-test, NUS-test and KPTimes-test datasets with similar number of keywords per document than its two train sets but does not explain the competitive performance of CatSeqD on the SemEval test set that has 6.71 present keywords per document. Even more importantly, it does not explain the large difference in performance between TNT-KID and CatSeqD on the JPTimes-test. This suggests that there is another factor influencing the performance of some keyword detectors.

The second hypothesis suggests that the difference in performance could be explained by the difference in training regimes and the different tactics used for keyword detection by the two systems. While TNT-KID is fine-tuned on each of the datasets, no fine-tuning is conducted for CatSeqD that needs to rely only on the information obtained during training on the large KP20k-train and KPTimes-train datasets. This information seems sufficient when CatSeqD is tested on datasets that contain similar keywords than the train sets. On the other hand, this training regime does not work for datasets that have less overlapping keywords.

Figure 4 supports this hypothesis by showing strong correlation between the difference in performance in terms of F@10 and the percentage of keywords that appear both in the CatSeqD train sets (KP20k-

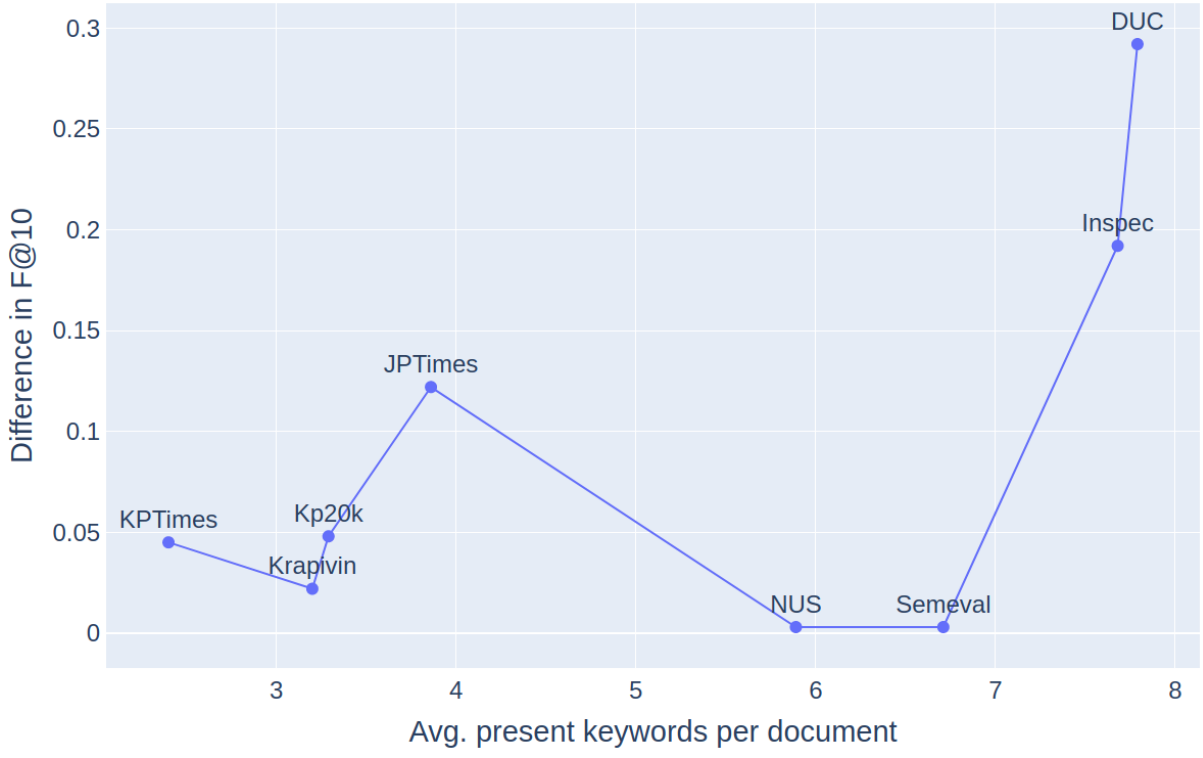


Figure 3: Relation between the average number of present keywords per document for each test dataset and the difference in performance ($F@10_{\text{TNT-KID}} - F@10_{\text{CatSeqD}}$).

Figure 4: Relation between the percentage of keywords that appear in the train set for each test dataset and the difference in performance ($F@10_{\text{TNT-KID}} - F@10_{\text{CatSeqD}}$).

train and KPTimes-train for computer science and news domain, respectively) and the test datasets. DUC and Inspec datasets have the smallest overlap, with only 17 percent of keywords in DUC appearing in the KPTimes-train and with 48 percent of keywords in Inspec appearing in the KP20k-train set. On the other hand, Krapivin, NUS, KP20k and KPTimes, the test sets on which CatSeqD performs more competitively, are the datasets with the biggest overlap, reaching up to 95 percent for KPTimes-test.

Figure 4 also explains a relatively bad performance of CatSeqD on the JPTimes corpus (see Table 2) despite the smaller average number of keywords per document. Interestingly, despite the fact that no dataset specific fine-tuning for TNT-KID is conducted on the JPTimes corpus (since there is no validation set available, fine-tuning is conducted on the KPTimes-valid), TNT-KID manages to outperform CatSeqD on this dataset by about 12 percentage points. This suggests that a smaller keyword overlap between train and test sets has less of an influence on the TNT-KID and could be explained with the fact, that CatSeqD considers keyword extraction as a generation task and tries to generate a correct keyword sequence, while TNT-KID only needs to tag an already existing word sequence, which is an easier problem that perhaps requires less specific information gained during training.

According to the Figure 4, the SemEval test set is again somewhat of an outlier. Despite the keyword overlap that is quite similar to the one in the JPTimes test set and despite having a relatively large set of present keywords per document, CatSeqD still performs competitively on this corpus. This points to a hypothesis that there might be another unidentified factor, either negatively influencing the performance of TNT-KID and positively influencing the performance of CatSeqD, or the other way around.

We also experimented with CatSeqD fine-tuning in order to allow fairer comparison of the results, but this did not result in improved performance of CatSeqD. As this is not the core part of this deliverable, we point the reader to the Section CatSeqD fine-tuning in the paper by Martinc, Škrlj, & Pollak (2020) in Appendix A.

2.4.2 Dissecting the attention space

One of the advantages of the transformer architecture is its employment of the attention mechanism, that can be analysed and visualized, offering insights into inner workings of the system and enabling interpretation of how the neural net tackles the keyword identification task. This insight is nevertheless limited, since it is still unclear what exactly is the nature of the relationship between attention weights and model outputs (Jain & Wallace, 2019). The TNT-KID attention mechanism consists of multiple attention heads (Vaswani et al., 2017) – square matrices linking pairs of tokens within a given text – and we explored how this (activated) weight space can be further inspected via visualization and used for interpretation.

While square attention matrices show importance of the correlations between all tokens in the document for a keyword identification task, we focused only on the diagonals of the matrices, which indicate how much attention the model pays to the "correlation" a specific word has with itself, i.e., how important is a specific word for the classification of a specific token as either being a keyword or not. We extracted these diagonal attention scores for eight attention heads of the last out of eight encoders, for each of the documents in the SemEval-test and averaged the scores across an entire dataset by summing together scores belonging to the same position in each head and dividing this sum with the number of documents. Figure 5 shows the average attention score of each of the eight attention heads for each token position. While there are distinct differences between heads, a distinct peak at the beginning of the attention graph can be observed for all heads but one (head 4), which means that heads generally pay more attention to the tokens at the beginning of the document. This suggests that the system has learned that tokens appearing at the beginning of the document are more likely to be keywords (Figure 6 shows the actual keyword count for each position in the SemEval corpus) and once again shows the importance of positional information for the task of keyword identification.

Another insight into how the system works can be gained by analysing how much attention was paid to each individual token in each document. Figure 7 displays attentions for individual tokens, as well as marks them based on predictions for an example document from the SemEval-test. Green tokens were correctly identified as keywords, red tokens were incorrectly identified as keywords and less transparency (more colour) indicates that a specific token received more attention from the classifier.

Figure 7 shows that, at least for this specific document, tokens that were either correctly or incorrectly classified as keywords did receive more attention than an average token. There are also some tokens that received a lot of attention and were not classified as keywords, e.g., *eos* (end of sentence signs) and *pad* (padding) signs, and also words like *of, is, we, etc.*. Another interesting thing to notice is the fact, that the amount of attention associated with individual tokens that appear more than once in the document varies and is somewhat dependent on the position of the token. [13]

2.5 Ablation study

In this section we explore the influence of several choices and building blocks of the keyword extraction workflow on the overall performance of the model:

- **Language model pretraining**; assessment whether pretraining positively affects the performance of the keyword extraction and if the improvements are dataset or domain specific.
- **Choice of pretraining regime**; comparison of two pretraining objectives, autoregressive language modelling and masked language modelling described in Section 2.2.2.
- **Choice of input tokenization scheme**; comparison of two tokenization schemes, word tokenization and Sentencepiece (Kudo & Richardson, 2018) byte-pair encoding.

[13] Note that Figure 7 is just a motivating example. A more thorough statistical analysis of much more than just one document would be required in order to draw proper conclusions about the behavior of the attention mechanism during keyword identification.

- **Part-of-speech(POS) tags**; assessment whether adding POS tags as an additional input improves the performance of the model.

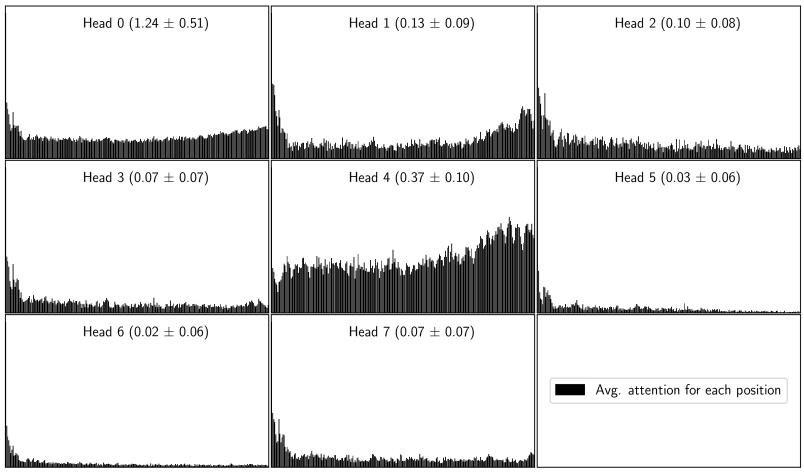


Figure 5: Average attention for each token position in the SemEval corpus across eight attention heads. Distinct peaks can be observed for tokens appearing at the beginning of the document in all but one out of eight attention heads.

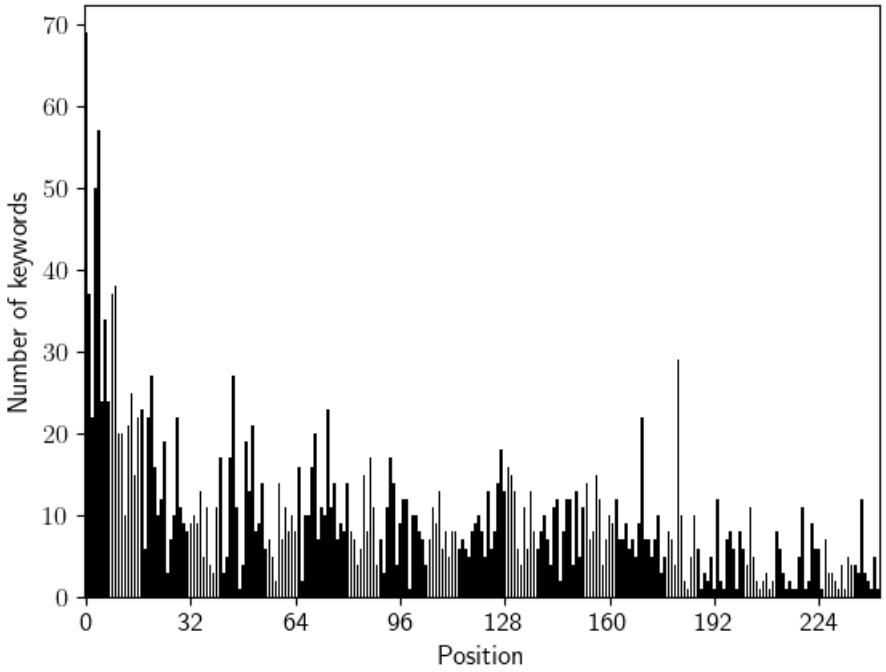


Figure 6: Number of keywords for each token position in the SemEval corpus. Distinct peaks can be observed for positions at the beginning of the document.

- **Transformer architecture adaptations**; as was explained in Section 2.2.1, in the fine-tuning stage we add an additional BiLSTM encoder to the output of the transformer encoder. We also experiment with the addition of the BiLSTM+CRF token classification head on top of the model, as was proposed in Sahrawat et al. (2020) and described in Section 2.2.1. Here we assess the influence of these additions on the performance of the model.

Table 3 presents the results on all datasets for several versions of the model, a model with no language model pretraining (*nolm*), a model pretrained with an autoregressive language model objective (*lm*), a model pretrained with a masked language model objective (*maskedlm*), a model pretrained with an autoregressive language model objective and leveraging byte-pair encoding tokenization scheme (*lm+bpe*), a model pretrained with an autoregressive language model objective and leveraging additional POS tag sequence input (*lm+pos*), a model pretrained with an autoregressive language model objective and a BiLSTM encoder (*lm+rnn*), a model pretrained with an autoregressive language model objective leveraging byte-pair encoding tokenization scheme and a BiLSTM encoder (*lm+bpe+rnn*), and a model pretrained with an autoregressive language model objective leveraging byte-pair encoding tokenization scheme and a BiLSTM+CRF token classification head (*lm+bpe+crf*) .

On average (see last two rows in Table 3), by far the biggest boost in performance is gained by employing the autoregressive language model pretraining (column *lm*), improving the F@5 score by about 10 percentage points and the F@10 score by 11 percentage points in comparison to no language model

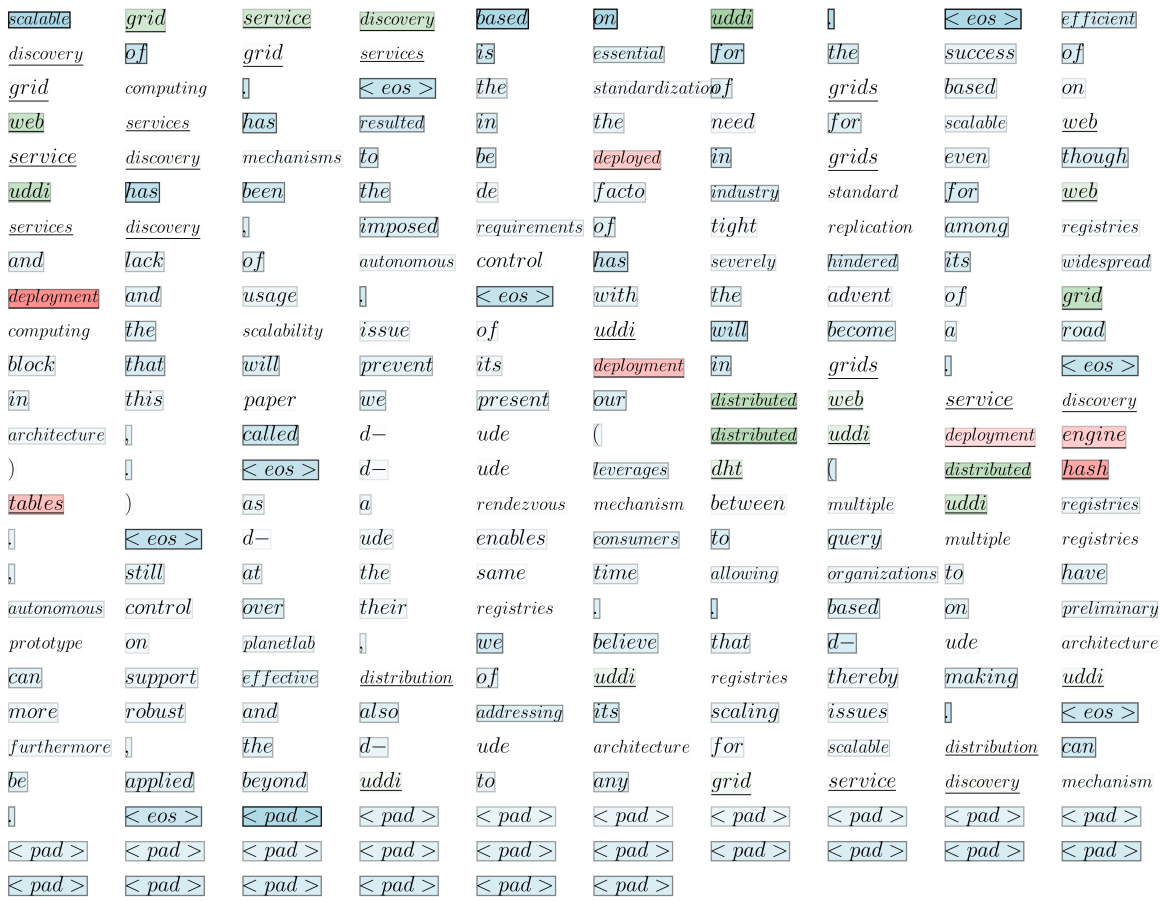


Figure 7: Attention-colored tokens. Green ones were correctly identified as keywords, red ones were incorrectly identified as keywords and less transparency indicates stronger attention for the token. Underlined words in italic were identified as keywords by the system.

Table 3: Results of the ablation study. Column *lm+bpe+rnn* represents the results for the model that was used for comparison with other methods from the related work in Section 2.3.3.

	nolm	lm	maskedlm	lm+bpe	lm+pos	lm+rnn	**lm+bpe+rnn**	lm+bpe+crf
					KP20k			
F1@5	0.2544	0.2922	0.2476	0.2958	0.3003	0.3349	0.3418	**0.3478**
F1@10	0.2304	0.2836	0.2313	0.2941	0.2986	0.3382	0.3457	**0.3521**
					Inspec			
F1@5	0.2868	0.4099	0.2875	0.4255	0.4136	**0.4506**	0.4471	0.4463
F1@10	0.3636	0.4994	0.3704	0.4871	0.5012	**0.5253**	0.5252	0.5147
					Krapivin			
F1@5	0.1919	0.2277	0.2046	0.2879	0.2494	0.3088	0.3009	**0.3142**
F1@10	0.1904	0.2314	0.2029	0.2884	0.2555	0.3164	0.3070	**0.3178**
					NUS			
F1@5	0.1909	0.3319	0.2372	0.3352	0.3339	0.3419	**0.3502**	0.3371
F1@10	0.1902	0.3492	0.2552	0.3586	0.3518	0.3626	**0.3686**	0.3658
					SemEval			
F1@5	0.1671	0.3070	0.1842	0.2462	0.2780	0.2696	**0.2921**	0.2524
F1@10	0.1950	0.3469	0.2528	0.2913	0.3426	0.3303	**0.3552**	0.3007
					KPTimes			
F1@5	0.2864	0.4242	0.3052	0.4211	0.4306	0.4627	**0.4691**	0.4408
F1@10	0.2760	0.4208	0.3017	0.4208	0.4300	0.4609	**0.4693**	0.4413
					JPTimes			
F1@5	0.2490	0.3305	0.2644	0.3341	0.3359	**0.3790**	0.3570	0.3357
F1@10	0.2478	0.3344	0.2705	0.3373	0.3402	**0.3823**	0.3596	0.3372
					DUC			
F1@5	0.1951	0.2848	0.1523	0.2759	0.2918	0.3003	**0.3115**	0.2943
F1@10	0.2265	0.3340	0.1979	0.3213	0.3386	0.3432	**0.3551**	0.3342
					Average			
F@5	0.2277	0.3260	0.2354	0.3277	0.3292	0.3560	**0.3587**	0.3461
F@10	0.2400	0.3500	0.2603	0.3499	0.3573	0.3824	**0.3857**	0.3705

pretraining (column *nolm*).

On the other hand, using the masked language modelling pretraining (column *maskedlm*) objective on average yields only a somewhat negligible improvement of about 0.8 percentage point in terms of F@5 score and a slightly bigger improvement of about 2 percentage points in terms of F@10 score in comparison to no language model pretraining. Next, the results show that adding POS tags as an additional input (column *lm+pos*) leads to only marginal performance improvements. Some previous studies suggest that transformer based models that employ transfer learning already capture sufficient amount of syntactic and other information about the composition of the text (Jawahar et al., 2019). Our results therefore support the hypothesis that additional POS tag inputs are somewhat unnecessary in the transfer learning setting but additional experiments would be needed to determine whether this is task/language specific or not.

Another adaptation that does not lead to any significant improvements when compared to the column *lm* is the usage of the byte-pair encoding scheme (column *lm+bpe*). Nevertheless, usage of byte-pair encoding does have an additional positive effect of drastically reducing the vocabulary of the model (e.g., for computer science articles, this means a reduction from about 250.000 tokens to about 30.000) and with it also the number of parameters in the model (from about 290 million to about 70 million). Furthermore, adding an additional BiLSTM encoder in the fine-tuning stage of a pretrained model (column *lm+rnn*) leads to consistent improvements on almost all datasets and to an average improvement of about 3 percentage points in terms of both F@5 and F@10 scores. This confirms the findings from the related work that recurrent neural networks work well for the keyword detection task and also explains why a majority of state-of-the-art keyword detection systems leverage recurrent layers. Last but not

least, we present results for a model in which we employed autoregressive language model pretraining, used byte-pair encoding scheme and added a BiLSTM encoder (column *lm+bpe+rnn*) that was used for comparison with other methods from the related work in Section 2.3.3, and results for the approach proposed by Sahrawat et al. (2020), where a BiLSTM+CRF token classification head is added on top of the transformer encoder, that employs byte-pair encoding scheme and autoregressive language model pretraining (column *lm+bpe+crf*). The BiLSTM+CRF performs quite well, outperforming all other configurations on two (KP20k and Krapivin) datasets. On average it however still performs by more than 1 percentage point worse than both configurations employing an added BiLSTM encoder. For more detailed analysis and interpretation of results for specific datasets, see the Ablation study section in Martinc, Škrlj, & Pollak (2020) in Appendix A.

As a result of this ablation study, the TNT-KID refers to the best settings, i.e. lm+bpe+rnn, for which the results are reported in Section 2.3.3 and is used also for training the media partners' models in Section 3.

2.6 Conclusion

We have presented TNT-KID, a novel transformer based neural tagger for keyword identification that leverages a transfer learning approach to enable robust keyword identification on a number of datasets. The presented results show that the proposed model offers a robust performance across a variety of datasets with manually labelled keywords from two different do-mains. We present the results of TNT-KID applied to the media partners' datasets in Seciton 3.

3 Keyword extraction experiments on media partners' datasets

TNT-KID was developed for the purpose of the real-world usage in the media environment, i.e. to automatize and accelerate the manual keyword tagging procedure conducted by the journalists for each newly produced news article. The idea is to employ TNT-KID as a keyword recommendation system, which would recommend keyphrase candidates that journalist could quickly pick out of the list of all recommended keywords for the use in production. For this reason, in this section we present the experiments in which we applied TNT-KID to media partners' data, in order to determine its performance in the real-life multilingual media company setting. So far, we have presented TNT-KID to ExM and Styria and both companies expressed very positive feedback and showed interest in testing the system for possible integration.

Since initial feedback from the ExM media house indicated that a general preference is to return a somewhat longer list of keywords then the one usually returned by TNT-KID, we also present an additional TF-IDF tagset matching technique that improves the recall of the proposed keyword extraction system. The ExM also expects that the system should return a constant number of keywords for each article and that this number should be large enough to offer a diverse spectrum of candidates, while still small enough to enable the journalist a quick overview and selection of appropriate candidates. For this reason, we decided that the system should return 10 candidates, which offers a reasonable trade-off between diversity of keywords and ease of selection. The new hybrid system first checks how many keywords were returned by TNT-KID and if the number is smaller than ten, the list is expanded by the best ranked keywords returned by the TF-IDF based extraction system, where only the keywords that are in the tagset are considered.

In Section 3.1 we present the TF-IDF tagset matching technique, media partner datasets are described in Section 3.2, while Sections 3.3 and 3.4 present the conducted experiments and results, respectively. The conclusions are given in Section 3.5.

3.1 TF-IDF based tagset matching

While the main feature of TNT-KID is that it is able to find new keywords, regardless of the fact if the words appeared in the training set or not, the TF-IDF tagset matching based method, which is less precise, is limited to only returning keyword candidates which have appeared as keywords in the training set or were pre-approved by the media house editors. These predetermined keyword sets contain 26,896 keywords for Croatian, 4,036 for Latvian, 5,953 for Russian and 59,242 for Estonian. For Croatian and Latvian, the tagset was constructed automatically, i.e. by taking all the keywords from the training set, while for Estonian and Russian, the tagset was an additional resource provided by the media partners.

First, each article in the dataset is preprocessed according to the pipeline described in Figure 8. The title and text of the article are concatenated and lowercased. Next, we conduct stopword removal and tokenization. We strip any punctuation from each token and lemmatize the sequence of tokens leveraging the Lemmagen (Juršič et al., 2010) lemmatizer, which supports languages for all datasets. The final cleaned textual representation consists of the concatenation of all of the preprocessed words from the document. We apply the same preprocessing procedure also on the predetermined keyword sets for each language.

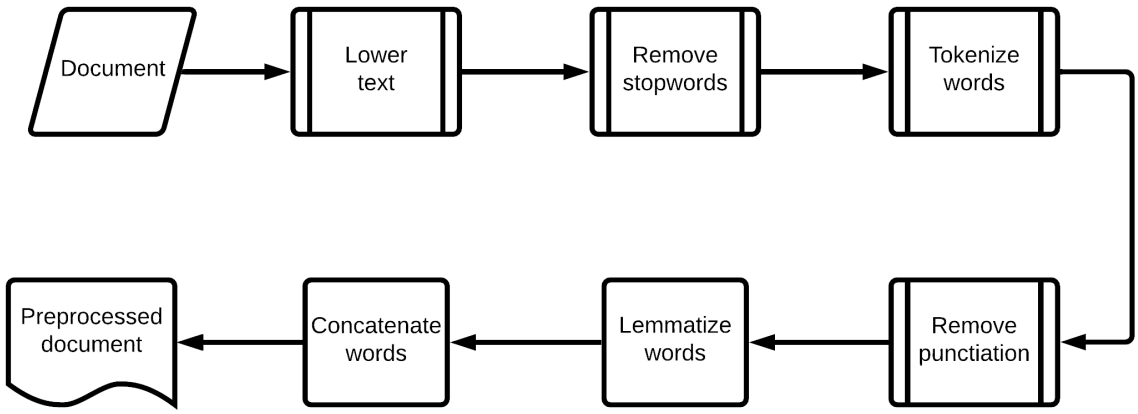


Figure 8: Preprocessing pipeline.

The TF-IDF weighting scheme assigns each word its weight $w_{i,j}$ based on the frequency of the word in the document (term frequency) and the number of documents the word appears in (inverse document frequency). More specifically:

1. *Term-frequency* (tf) counts the number of appearances of a word in the document.
2. *Inverse-document-frequency* (idf) ensures that words appearing in more documents are assigned lower weights and is calculated as:
$$\log_e(\frac{|D|}{df_i})$$
where df_i is the number of documents that contain word i and $|D|$ the number of documents in the corpus.

The entire metric is calculated as:
$$TF - IDF_i = tf_{i,j} \cdot \log_e(\frac{|D|}{df_i})$$
where $tf_{i,j}$ is the number of occurrences of the word i in the document j, df_i is the number of documents containing word i and $|D|$ the number of documents.

The assumption is that words with a higher TF-IDF value are more likely to be keywords. Nevertheless, since the results in Section 2.3.3 indicate that unsupervised approaches towards keyword extraction tend to produce much worse results than supervised, the list of TF-IDF ranked keyword candidates obtained for each document is limited only to words which appeared in predefined keyword tagsets.

For combining the TNT-KID and TF-IDF (TNT-KID+TF-IDF), we first take the keywords by TNT-KID, while the missing keywords (to achieve the set goal of 10) are selected by taking the top-ranked candidates from the TF-IDF tagset matching extraction.

3.2 Media partners datasets

We conduct experiments on datasets containing news in four languages, Latvian, Estonian, Russian and Croatian. Latvian, Estonian and Russian datasets contain news from Ekspress Group, specifically from our Estonian partner Ekspress Meedia (ExM) and from Latvian Delfi. The Croatian dataset was acquired from 24sata news portal belonging to Styria Media Group, one of the leading media groups in Austria, Croatia, and Slovenia. Most of the content in the dataset comes from two leading news portals in the Croatian market in terms of page visits and business results. The dataset statistics are presented in Table 4. For the training set, we used the articles from 2018, while for the test set the articles from 2019 were used.

Table 4: Media partners' datasets used for empirical evaluation of keyword extraction algorithms. *No.docs* stands for number of documents, *Avg. doc. length* stands for average document length in the corpus, *Avg. kw.* stands for average number of keywords per document in the corpus, *% present kw.* stands for the percentage of keywords that appear Avgin the corpus (i.e., percentage of document's keywords that appear in the text of the document) and *Avg. present kw.* stands for the average number of keywords per document that actually appear in the text of the specific document.

Dataset	No. docs	Avg. doc. length	Avg. kw.	% present kw.	Avg. present kw.
latvian-train	13133	378.03	3.23	0.53	1.69
latvian-test	11641	460.15	3.19	0.55	1.71
estonian-train	10750	395.24	3.81	0.65	2.77
estonian-test	7747	411.59	4.09	0.69	3.12
croatian-train	47479	420.32	3.10	0.47	1.32
croatian-test	5277	464.14	3.28	0.55	1.62
russian-train	13831	392.82	5.66	0.76	4.44
russian-test	11475	335.93	5.43	0.79	4.33

3.3 Experimental setup

We conducted experiments on the datasets described in Section 3.2. We evaluated TNT-KID, the TF-IDF tagset matching, as well as the combined approach. In addition, for Croatian and Estonian, we also present three strongest baselines employed in the English TNT-KID study, CopyRNN (Meng et al., 2017), CatSeqD (Yuan et al., 2019) and BERT + BiLSTM-CRF (Sahrawat et al., 2020), which were trained on the same train datasets as TNT-KID. For TNT-KID and the baselines, we employ the same hyperparameter configurations that were used for the English experiments (see Section 2.3).

- **TNT-KID**: The best configuration of our proposed TNT-KID approach (a model pretrained with an autoregressive language model objective, leveraging Sentencepiece (Kudo & Richardson, 2018) byte-pair encoding tokenization scheme and an additional BiLSTM encoder) was used for keyword extraction and the input text was lowercased for all languages. Same hyperparameters were used as for the experiments on the English datasets (see Section 2.3).

- **TF-IDF tagset matching:** We use TF-IDF-based weighting of keywords and select the top-ranked keywords that are present in the tagset (see Section 3.1). The preprocessing procedure is described in Figure 8.
- **TNT-KID + TF-IDF:** first the keywords are extracted by TNT-KID, and then if less then 10 keywords are selected, the additional keywords are taken from the top ranked TF-IDF tagset matching approach.
- **CopyRNN:** The best performing model from Meng et al. (2019) is used, reffered to as CopyRNN.
- **CatSeqD:** An improved one-to-seq approach by (Yuan et al., 2019), who incorporated the mechanisms called *semantic coverage* and *orthogonal regularization* to constrain the over-all inner representation of a generated keyword sequence to be semantically similar to the overall meaning of the source text and therefore force the model to produce diverse keywords.
- **BERT + BiLSTM-CRF**(Sahrawat et al., 2020): instead of using a pretrained GPT-2 with a BiLSTM-CRF token classification head, as we do for the English experiments, here we use a pretrained uncased multilingual BERT with an embedding size of 768 and 12 attention heads (Devlin et al., 2018)[14], since there is no multilingual GPT-2 model available.

For TNT-KID, which is the only model that requires language model pretraining, language models were trained on train sets in Table 4 for up to ten epochs. Next, TNT-KID and BERT + BiLSTM-CRF were fine-tuned on the training datasets, which were randomly split into 80 percent of documents used for training and 20 percent of documents used for validation. The documents containing more than 256 tokens are truncated, while the documents containing less than 256 tokens are padded with a special $< pad >$ token at the end. Same as for the experiments described in Section 2.3, we fine-tuned each model for a maximum of 10 epochs and after each epoch the trained model was tested on the documents chosen for validation. The model that showed the best performance on this set of validation documents (in terms of F@10 score) was used for keyword detection on the test set.

For CopyRNN and CatSeqD, we employ the training procedure recommended by the authors of the systems, i.e., the models are trained for 100.000 train steps on all train sets using the default configuration.

For evaluation, we present precision, recall and F1 score, while F1@10 and R@10 are the most relevant metrics for our media partners. Only keywords which appear in a text (present keywords) were used as a gold standard since we only evaluate approaches for keyword tagging that are not capable of finding keywords which do not appear in the text. Lowercasing and lemmatization (using Lemmagen lemmatizer (Juršič et al., 2010)) are performed on both the gold standard and the generated keywords (keyphrases) during the evaluation.

3.4 Keyword extraction results

The results for TNT-KID, TF-IDF and the combination of both for all media datasets are presented in Table 5, as well as additional results with other state-of-the-art approaches for Croatian and Estonian (which are the main two languages, as they cover the core needs of EMBEDDIA partners).

The results vary from one dataset to another. While the best performance in terms of F1@5 and F1@10 on the Croatian dataset is achieved by BERT + BiLSTM-CRF, on the Estonian dataset TNT-KID offers more competitive performance. The other two baselines, CopyRNN and CatSeqD perform somewhat uncompetitively on both Croatian and Estonian. Interestingly, CopyRNN manages to outperfom CatSeqD on both datasets, even though CatSeqD offered more competitive performance than CopyRNN on English.

TF-IDF on itself performs poorly on all datasets besides Croatian, where it offers performance comparable to CopyRNN and CatSeqD. The most likely reason for this discrepancy is the difference in quality of

[14]We use the implementation of BERT from the Transformers library (`https://github.com/huggingface/transformers`).

Table 5: Results on the media partner datasets.

Model	P@5	R@5	F1@5	P@10	R@10	F1@10
Croatian						
TF-IDF	0.1518	0.3404	0.2100	0.1289	0.5607	0.2096
TNT-KID	0.3485	0.5359	0.4223	0.3354	0.5594	0.4194
TNT-KID + TF-IDF	0.2793	**0.6517**	0.3911	0.2034	**0.9230**	0.3334
CopyRNN	0.2277	0.3166	0.2418	0.2263	0.3193	0.2409
CatSeqD	0.1580	0.3561	0.2040	0.1389	0.4052	0.1887
BERT + BiLSTM-CRF	**0.4728**	0.4585	**0.4655**	**0.4724**	0.4602	**0.4662**
Estonian						
TF-IDF	0.0377	0.0785	0.0510	0.0388	0.1523	0.0619
TNT-KID	0.5067	0.5649	**0.5343**	0.5055	0.6035	**0.5502**
TNT-KID + TF-IDF	0.2956	**0.5924**	0.3944	0.1864	**0.7061**	0.2949
CopyRNN	0.4706	0.3517	0.3611	0.4703	0.3523	0.3611
CatSeqD	0.3650	0.3910	0.3332	0.3515	0.4037	0.3271
BERT + BiLSTM-CRF	**0.5221**	0.4528	0.4850	**0.5199**	0.4681	0.4927
Russian						
TF-IDF	0.0822	0.1086	0.0936	0.0817	0.1686	0.1101
TNT-KID	**0.6896**	0.5906	**0.6363**	**0.6897**	0.6196	**0.6528**
TNT-KID + TF-IDF	0.4329	**0.6384**	0.5160	0.2932	**0.7468**	0.4211
Latvian						
TF-IDF	0.0518	0.1036	0.0690	0.0419	0.1417	0.0647
TNT-KID	**0.3718**	**0.4120**	**0.3909**	**0.3715**	0.4208	**0.3946**
TNT-KID + TF-IDF	0.1415	0.3417	0.2001	0.1230	**0.5089**	0.1982

the pre-approved keyword sets given to us by the media partners, since the manual inspection revealed that the keyword sets given to us by the Express media contain some noise, such as grammatical mistakes and keywords of doubtful quality. On the other hand, a manual inspection of the Croatian keyword set revealed no such problems.

Nevertheless, combining TNT-KID and TF-IDF does improves recall@5 and recall@10 on all datasets, since the combination of the systems in a large majority of cases returns more keywords than the TNT-KID itself. While the best recall@10 is achieved for Croatian (92.30%), considerable recall improvements can also be observed for Russian (74.68%) and Estonian (70.61%). On the latter two datasets TNT-KID achieves the best F1 scores, about 55% and about 65% for Estonian and Russian, respectively. On the other hand, the combination of TNT-KID and TF-IDF returns uncompetitive F1 scores for all datasets. This is not surprising, since the combination always returns 10 keywords, i.e., much more than the average number of present gold standard keywords in the media partner datasets (see Table 4), which badly affects the precision of the approach.

3.5 Conclusions on experiments on media partners' datasets

In this section we have presented the evaluation of our TNT-KID on media partners' dataset, adaptation of the method (TNT-KID + TF-IDF) for improving recall, as well as several other evaluations of other state-of-the-art systems. Given that keyword assignment represents one of the main needs and possibilities for actual deployment in the media partners' settings, we remain focused on the monolingual settings and achieving best results, however in future cross-lingual evaluation might be of interest as a

scientific task.

4 TEXTA Hybrid keyword tagger

This section described the work by TEXTA, where the task was to develop a fast keyword classifier on a dataset by a media company Õhtuleht Newspaper Dataset. The work was done in collaboration with T6.2 and T6.3., as it is an integrated component of the Texta Toolkit part of the EMBEDDIA Media Assistant (EMA). The work is therefore also referred in the Deliverables D6.7 and D6.8. The method and experiments are in detail described in Vaik et al. (2020) (a paper that was published in the Industry track of the LREC conference), which can be found in Appendix A of Deliverable D6.8. In contrast to TNT-KID (Section 2) where the system considers all words in the article as a possible keyword, this section addresses the setting, where possible keywords are limited to a predefined tagset. In this sense, the task is similar to the TF-IDF extension of TNT-KID presented in Section 3 with the difference that in this work the solution proposed is resulting from a classifier (distinguishing between tags assigned to similar documents), and can therefore match also the keyword tags that are not present in the article.

The presented approach is a solution for multi-label classification with a massive label collection. The proposed approach incorporates a large number of binary classification models with label pre-filtering and employs methods and technologies shown to be applicable also in settings with limited high-end computational hardware. The system is evaluated on an Estonian newspaper article dataset which contains almost 2000 unique labels and has shown to perform over 80 times faster than applying all the binary models of the entire label set without negative impact on prediction scores. From EMBEDDIA partners, STT is having a similar setting, where they assign IPTC tags (from a tagset) to the articles. This approach can therefore in future experiments tested on their data and compared to computationally more demanding neural models.

4.1 Workflow of Hybrid Tagger

Hybrid Tagger (HT) incorporates a high number of binary classification models combined with unsupervised label pre-filtering in order to achieve real-time predictions for thousands of labels. HT uses the logistic regression classification algorithm from scikit-learn (Pedregosa et al., 2011), while the unsupervised pre-filtering is achieved by using Elasticsearch[15] engine's document retrieval features.

Elasticsearch is used because of its stable and scalable platform for retrieving documents and performing document similarity queries for the label pre-filtering. Elasticsearch also allows to disregard complex matrix computations and instead offers a fairly transparent way to filter labels based on the document similarity they have been assigned to.

This section describes preprocessing (Section 4.1.1), training (Section 4.1.2) and prediction (Section 4.1.3).

4.1.1 Preprocessing

The preprocessing pipeline consists of tokenization, lemmatization, part of speech (POS) tagging, and named entity recognition (NER). This is done by using TEXTA Multilingual Processor (MLP) which uses NLTK (Bird et al., 2009) with Stanford models and EstNLTK (Orasmaa et al., 2016).

[15] https://www.elastic.co/elasticsearch

4.1.2 Training

Binary classifiers used in HT can be trained on any text segment, e.g., title, content, author, etc. In our experimental setup, models are usually trained using lemmatized content and also optional POS tags. For vocabulary reduction, stop words are removed from all texts prior to training.

Training data is selected according to the pre-existing labelling. For each label, all existing *positive* examples (texts annotated with the specific label) and the same amount of randomly selected *negative* examples (texts not annotated with the specific label) are retrieved. Examples for all labels are then randomly split into training and validation sets (default 80-20).

In real-life scenarios, the training process may result in thousands of classification models which have a significant memory imprint when combined. To combat this problem, HashingVectorizer (Pedregosa et al., 2011) is used to vectorize the training data. It has significantly smaller memory imprint than other vectorizers supported by scikit-learn (e.g., commonly used TfIdfVectorizer).

For training models, 5-fold cross-validation and grid search are used to set the parameters, such as minimum and maximum length of *n*-grams, choice between word or character n-grams, value of C (inverse regularization parameter for logistic regression classifier). The best model from grid search is then validated using the validation set.

4.1.3 Prediction

Instead of applying all possible binary models, in this approach first a subset of models which most likely to provide the correct prediction are identified. This is done by finding *n* (default *n*=10) similar texts from the training data indexed in Elasticsearch and finding *m* (default *m*=10) most frequent labels assigned to the texts. Retrieving similar texts is done by using an Elasticsearch *more like this* query[16] which calculates top *k* words with the highest TF-IDF score per text and afterwards performs a disjunctive query using the pre-existing labels to match similar texts.

The prediction pipeline is as follows:

- preprocess the input text (optional);
- find *n* similar texts indexed in Elasticsearch;
- find top *m* labels attached to the texts found in the previous step;
- apply named entity recognition on the input to identify *l* entity-related labels and remove these binary models from the list of *m* models which will be used later for label prediction;
- retrieve all models for each of *m*-*l* top labels;
- apply models, retrieve and combine the list of predicted labels classified by the binary models and the entity-related labels, and output the results.

4.2 Evaluation

The evaluation shows that Hybrid Tagger performs significantly faster than the baseline model without negative impact on prediction scores. This is shown in detail in our paper (Vaik et al., 2020) available in Appendix of D6.8. Here, we omit the analysis in terms of the applicability of Hybrid Tagger depending on the available computation power and number of labels present in the dataset.

In this section we present the evaluation of the tagger in the case Study on Õhtuleht Newspaper Dataset. Õhtuleht dataset contains newspaper articles spanning from years 2013 to 2019, covering a wide range of topics (news, sports, entertainment, crime, etc). For evaluating HT, Õhtuleht dataset is split into train

[16] https://www.elastic.co/guide/en/elasticsearch/reference/current/query-dsl-mlt-query.html

and test set (100 000 and 2450, respectively), and trained 1870 binary classifiers on lemmatized articles' content using logistic regression as the predictor function. The minimum number of examples per each label was set to 50, resulting in disqualifying 108 labels with a smaller number of examples.

The results are provided in terms of average prediction time, precision, recall, F1 score and the number of predicted labels on the test data by applying:

1. Baseline model (BL) consisting of all binary classifiers;
2. Hybrid Tagger with the best detected parameter configuration (*n label candidates = 10*, *n similar texts = 10*) with NER enabled (HT NER) and NER disabled (HT).

Table 6: Comparing Baseline with Hybrid Tagger

	BL	HT NER	HT
n taggers	1870	10	10
n similar texts	/	10	10
n cores	24	24	24
NER enabled	no	yes	no
Time (s)	82.34	1.01	1.01
Precision	0.07	0.70	0.71
Recall	0.85	0.92	0.75
F1 score	0.12	0.76	0.67
n predicted labels	128.87	6.21	4.89

Table 6 gives an overview of Hybrid Tagger's performance compared with the baseline model (BL).

BL model's actual average prediction time is 82.34 seconds while HT labels one document on average with 1.01 seconds resulting in an actual speedup pf 81.5. The experimental results prove that HT performs significantly faster than the BL model.

Furthermore, BL's F1 score is only 0.12 as a result of extreme over labelling causing a very low precision of 0.07. HT's recall without NER is 0.75 being slightly lower than BL's average of 0.85. However, HT with NER obtains even better recall than the BL model with an average score of 0.92. It is still important to keep in mind that the applicability of NER is dataset-specific. r For more details the reader is referred to the paper in Appendix A in deliverable D6.8.

4.3 Conclusion on TEXTA Hybrid Tagger

This section presented an industry-driven solution, Hybrid Tagger, for multi-label classification with a large volume of keywords. The proposed system incorporates a high number of binary classification models coupled with unsupervised label pre-filtering and named entity recognition to achieve real-time predictions with thousands of labels. As the development of Hybrid Tagger was industry-driven, the time aspect is the main focus of the evaluation (see paper in D6.8).

The evaluation of Hybrid Tagger on Õhtuleht newspaper dataset shows that Hybrid Tagger helps to significantly improve both the prediction times and precision scores in comparison to executing all binary classification models of the label set. In future, we plan to test the approach on STT tags using IPTC classification (a standardised set of tags (keywords) used by news industry).

For now TEXTA Hybrid Tagger and TNT-KID were not compared, as they are applicable to different settings: while TNT-KID was developed for standard keyword extraction task, where any word (or multi-word expression) in the article can be a keyword, TEXTA Hybrid Tagger was applied in a special multi-label classification setting (where an article gets assigned a keyword from cca. 2000 possible labels, where there is no pre-condition that a label is also a word in the article). As the dataset used in experiments in this section is external to EMBEDDIA, we also did not use it for more detailed investigation whether the

labels correspond also to keywords in the article, which would allow for comparison of the approaches. In future, we will analyse the EMBEDDIA Finnish press agency dataset (STT), which is a similar setting to the one presented here, and in case the labels correspond to the words as used in the articles, we will be able to compare the two systems.

5 Keyword and term alignment

Term alignment refers to the process of aligning terms between two candidate term lists in two (or more) languages. The task originates from terminology and translation research fields. In the scope of EMBEDDIA, we consider the same techniques for aligning tagsets of keywords (i.e. the predefined lists from which journalist choose the keywords for their articles) in two different languages.

In previous deliverable D2.3, we described our replication study (published in Repar et al. (2019)) on bilingual term-alignment approach by Aker et al. (2013) and its adaptation. In this study, the term alignment is considered as a bilingual classification task: for each term pair, various features are created based on word dictionaries (i.e. features based on Giza++ word alignements) and word similarities (i.e. cognate-based features) across languages. In our replication study, we first showed that our reimplementation results were drastically lower than the ones' of original authors. However, after several adaptations, we managed to achieve competitive results.

In this delvierable, we first present how we used the method for aligning the tagset of keywords for Ekspress Meedia (ExM) (Section 5.1), and next how we adapted the method by considering features derived from cross-lingual word embeddings (Section 5.2).

5.1 ExM keyword tagset alignment

We present the experiments of using the system from Repar et al. (2019) for aligning the tagset of keywords in Russian and Estonian of ExM media partners. ExM have a large dataset of manually assigned keywords in two languages: Estonian and Russian. The dataset is skewed towards Estonian, but there is still a sizeable chunk of Russian keywords available for alignment. The dataset was split into two parts based on script type and we assumed Latin script words were Estonian and Cyrillic script words were Russian. Out of the total 65 830 keywords in the dataset, 59 632 were Estonian and 6 198 were Russian. The keywords in the two languages were not aligned and the task was as follows: for every Russian keyword, try to find an equivalent keyword in Estonian. Having this functionality would allow ExM to select a Russian or Estonian keyword and filter articles on the topic not just in the language of the keyword, but also in the language of the equivalent keyword found by our algorithm.

The term alignment approach described in (Repar et al., 2019) requires the following resources to work:

- an aligned list of terms (e.g., a bilingual dictionary)
- Giza++ word alignment dictionary or a large parallel corpus to calculate such word alignment dictionaries
- transliteration rules for characters not appearing in the opposite language

Few bilingual resources exist for this language pair which meant that we had to be flexible in terms of resources, even if they were not of the highest quality. For example, the Eurovoc dictionaries nor the parallel DGT corpus or the derived Giza++ translation tables were not available for Russian, as Russian is not one of the EU languages. Therefore, we used the following resources:

- **Aligned list of terms**: we used the General Multilingual Environmental Thesaurus[17] (GEMET), since

[17] https://www.eionet.europa.eu/gemet/en/about/

this was one of the few bilingual terminological resources that contained both Russian and Estonian.

- **Giza++ alignments**: Giza++ word alignment dictionaries were calculated from the OpenSubtitles2018 corpus available at OPUS portal[18].
- **Transliteration**: we used the transliterate Python package[19] for transliteration from Latin to Cyrillic scripts.

These resources were used for generating the features for training a machine-learning classification model as in Repar et al. (2019).

The created machine-learning model was then applied to align the Russian and Estonian keywords from the ExM tagset. The algorithm classified 4989 Estonian-Russian pairs as positive. A subset of 500 pairs were manually evaluated by a domain expert from ExM, proficient in both languages, who found that 74% of the positively classified pairs were correct keyword pairs.

5.2 Term alignment with novel embeddings features

In this section, we present experiments on term alignment, where we take our reimplementation of the bilingual term-alignment approach by Aker et al. (2013) as presented in (Repar et al., 2019) and consider additional embeddings-based features. More specifically, we wanted to explore if embeddings-based features can improve the overall performance of the term alignment algorithm as well as reduce the reliance on bilingual resources which can be scarce for language pairs not involving English as seen in Section 5.1. This work was done in collaboration between JSI and UL-CS.

As in the original experiments, we take the Eurovoc thesaurus (Steinberger et al., 2002) and in this deliverable we focus only on English and Slovene language pair.

We used the VecMap tool by (Artetxe et al., 2018) to align FastText embeddings of words that appear in the terms between the source and the target language and calculated cosine similarities between all pairs of source and target words. Following the feature generation procedure in (Repar et al., 2019), we generated the following new features (where instead of Giza++ features we use the embedding alignment based features):

- isFirstWordFirstMatch,
- isLastWordFirstMatch,
- percentageOfFirstMatchWords,
- percentageOfNotFirstMatchWords,
- longestFirstMatchedUnitinPercentage,

where *FirstMatch* means that the target term word (i.e. the first word in the target term) is also the most similar target word for a given source word based on cosine similarity of their FastText embeddings. For example, for Slovenian term *uničenje*, top aligned English word embeddings (and their cosine distance) are: *destruction (0.761), damage (0.586), confiscation (0.576)*, and the top ranked word (destruction) is the correct term translation.

Furthermore, we noticed that for a minority of source words, the target term word of a given source word does not appear at the top of the list of most similar words according to FastText cosine similarity, but it may appear in second or third place. To cover such cases, we generated 5 more features:

- isFirstWordTopNMatch,
- isLastWordTopNMatch,

[18] http://opus.nlpl.eu/OpenSubtitles-v2018.php
[19] https://pypi.org/project/transliterate/

- percentageOfTopNMatchWords,
- percentageOfNotTopNMatchWords,
- longestTopNMatchUnitinPercentage,

where *TopNMatch* means that the target term word can be found in the list of top n similar target words based on cosine similarity of their FastText embeddings (we selected n=3). For Slovenian term *demografska analiza* and the English *demographic analysis*, the feature *isFirstWordFirstMatch* is 0 (as the top three 3 embeddings alignments for *demografska* are *demography (0.6665), economic (0.5933), demographic (0.59*)), however the feature isFirstWordTopNMatch is 1, as *demographic* occurs in the 3rd place.

In total, 20 new features were generated (10 for each language direction, i.e. source to target and target to source). We added the new features to the training and test dataset and repeated the experiments with configurations 5, 6, 7, 8 and 10 from Repar et al. (2019) (see Table 3). For configuration 5, we simply added the new embedding features to the data sets[20] to see whether the new features improve the results while for the other 4 configurations, we also replaced the Giza++ features with embedding features to evaluate if the FastText features can effectively replace them to reduce the reliance on large parallel corpora.

Below is a brief explanation of the experiments from (Repar et al., 2019) that we reevaluated for this deliverable:

- **unbalanced train set**: the ratio of positive and negative terms in the training set is 1:200 (Configuration 5 in Repar et al. (2019))
- **train set filtering 1**: the train set is filtered according to a qualitative analysis of the features and the positive/negative ratio is 1:1 (Configuration 6 in Repar et al. (2019))
- **train set filtering 2**: the train set is filtered according to a qualitative analysis of the features and the positive/negative ratio is 1:10 (Configuration 7 in Repar et al. (2019))
- **train set filtering 3**: the train set is filtered according to a qualitative analysis of the features and the positive/negative ratio is 1:200 (Configuration 8 in Repar et al. (2019))
- **cognates**: the train set is filtered according to a qualitative analysis of the features with a focus on cognate term candidates and the positive/negative ratio is 1:200 (Configuration 10 in Repar et al. (2019))

Results in Table 7 indicate that in the most initial setting (on unbalanced train set, Configuration nb. 5 in our experiments in Repar et al. (2019)) adding the new embeddings features improve the performance: it is clearly shown that adding embeddings features results in a significant improvement of precision and F1 score at a cost of somewhat lower recall. However, in additional settings (using training set filtering), the improvement is not straightforward and more investigation is needed to discover the best feature combination. Configuration 10 looks promising because using just embeddings features (as well as cognate features which also do not require large parallel corpora), we were able to achieve high precision (overall the best precision of all the experiments).

In addition to FastText embeddings, we also performed experiments with ELMo contextual embeddings, however the cross-lingual alignments were not of sufficient quality to be of use. In future, we plan to evaluate the FastText features on the ExM keyword alignment experiments presented in Section 5.1.

[20] In total we have 28 features described in Repar et al. (2019) (Giza++ and cognate based)—13 dictionary-based, 5 cognate-based features with transliteration rules (in these experiments these were calculated only for one direction, while cognate based features without transliterations were not considered) and 10 combined features—and 20 novel embedding-based features).

Config (config. number from Repar et al. (2019))		precision	recall	F1 score
unbalanced train set (nb. 5)	giza	0.4299	**0.7617**	0.5496
unbalanced train set (nb. 5)	combined	0.5375	0.6800	0.6004
train set filtering 1 (nb. 6)	giza	0.5969	0.6417	0.6185
train set filtering 1 (nb. 6)	embeddings	0.1845	0.5783	0.2798
train set filtering 1(nb. 6)	combined	0.1583	0.7100	0.2589
train set filtering 2 (nb. 7)	giza	0.9042	0.5350	**0.6723**
train set filtering 2 (nb. 7)	embeddings	0.6487	0.3017	0.4118
train set filtering 2 (nb. 7)	combined	0.5333	0.6400	0.5818
train set filtering 3 (nb. 8)	giza	0.9342	0.4966	0.6485
train set filtering 3 (nb. 8)	embeddings	0.9545	0.2450	0.3899
train set filtering 3 (nb. 8)	combined	0.8170	0.5133	0.6305
cognates (nb. 10)	giza	0.8732	0.5167	0.6492
cognates (nb. 10)	embeddings	**0.9618**	0.3617	0.5242
cognates (nb. 10)	combined	0.8991	0.5200	0.6589

Table 7: Results of additional experiments. *giza* lines contain results from the previous experiments described in (Repar et al., 2019), *embeddings* lines contain results from experiments where we replaced Giza++ features with embeddings features, and *combined* lines contain results where we used both Giza++ and FastText embeddings alignment features.

6 Additional experiments on term extraction

Already in the introduction, we have explained that the two domains (terminology and keyword extraction) are highly related and offer many opportunities for joint exploitation. While keywords are important descriptors of single documents, terms usually refer to domain-specific expressions and are extracted from a collection of documents. While in the previous section (Section 5) we have shown how methods from terminology science can directly benefit the keyword matching task in media setting, in this section we show how EMBEDDIA contextual embeddings and techniques can be applied to a terminology extraction task (Section 6.1). Next, we describe a novel canonical form generator for Slovene (Section 6.2), which is needed for transforming terms as well as keywords to their canonical forms, which in morphologically-rich Slavic languages differs from the lemmatised form. Finally, we describe a method for extracting terminological semantic relations using intersections of embeddings (Section 6.3), which has potential for finding lexical variation in natural language generation.

6.1 Term extraction using contextual embeddings

In this section, we present our first attempts in developing a term extraction system based on ELMo contextual embeddings. This work was done in collaboration between JSI and UL-CS.

We believe that at its core, the issue of terminology extraction is one of context. Terms are linguistic representations of domain-specific concepts implying that the same linguistic representation has a different meaning and/or behaves differently in a domain-specific setting than it would in the general language. For example, contrastive approaches to terminology extraction, such as the ones by Vintar (2010); Pollak et al. (2012), are based on comparison of the frequencies between the domain and general corpus.

In terms of representing context, a major research development in natural language processing has been the introduction of contextual word embedding models (Peters et al., 2018a; Devlin et al., 2018) in recent years. In contrast to the static word embedding models, e.g., (Mikolov et al., 2013), where during training all senses of a given word contribute relevant information in proportion to their frequency in the training corpus (which causes the final word embedding to be placed somewhere in the weighted middle of all words' meanings), the contextual embeddings generate a different vector for each context a word

appears in, with this context typically being defined sentence-wise. Static embeddings can therefore not handle well the representation of polysemous words, where for example, the word embedding of the word "bank", would include (at least) two meanings: 1) a financial institution and 2) a river bank. On the other hand, contextual word embedding models allow researchers to get word embeddings for words in their (domain-)specific context which is what we aim to exploit further in our research of domain-specific terminology. For more details on differences between static and contextual embeddings see deliverables D1.2 and D1.3.

We have devised an initial term extraction approach utilizing contextual word embeddings by comparing the term vectors in a general language corpus and a domain-specific corpus. In this regard, our research follows other similar approaches comparing general and domain corpora (Vintar, 2010; Pollak et al., 2012), but instead of comparing relative frequencies, we compare word embeddings, with the idea that (single or multi-word) expressions with a large distance between the general domain and specific domain vectors are the most likely to be terms. The system consists of four phases: 1) term candidate extraction using traditional methods with part-of-speech patterns, 2) calculation of average lemma embeddings in both corpora, 3) calculation of term embeddings in both corpora and 4) cosine similarity comparison of term embeddings in general and domain corpus.

The experiments were performed on the following corpora: the Slovenian corpus of karstology (Vintar et al., 2019), where we have previously evaluated term extraction methods (Pollak et al., 2019) and the English corpus in the ACTER dataset (Terryn et al., 2019), which is one of the rare multilingual gold standard corpora for terminology.

The approach was applied as follows:

Step 1: Term candidate extraction. We start by lemmatizing and tagging the domain-specific corpus (Karst for Slovenian, ACTER for English) and then use a set of pre-defined part-of-speech patterns, such as noun+noun or adjective+noun, to collect the initial term candidates. This step is similar to other previous approaches.

Step 2: Calculation of average lemma embeddings in both corpora For the general domain corpus embeddings, we use the ELMo embeddings produced from the reference corpus of Slovenian language Gigafida 2.0 (Krek et al., 2020), while for English, we used the models trained on the English Wikipedia. The corpora are not lemmatized and the contextual embeddings are calculated for each word form separately. To counter this, we calculate a lemma average of each word by lemmatizing all words, summing up all word form embeddings and dividing them by the number of different word forms[21]. Then we pass all sentences in the domain-specific corpus one by one to the ELMo model to receive contextual embeddings of each word in each sentence and calculate word averages. To follow the same approach as for the general corpus, we don't lemmatize the domain-specific corpus, but rather calculate lemma averages as above.

Step 3 and 4: Calculation of term embeddings in domain and reference corpus and ranking of term candidates by term embeddings comparison. For each term candidate from the initial list from step 1, we find its average embedding, where single word terms correspond to average embeddings of the lemma in Step 2, while for multi-word terms, we calculate term embeddings as the average of all its constituent words (lemmas). We calculate term embeddings in both (domain and general) corpora and calculate cosine similarity between them and finally rank the terms according to the cosine similarity of their vectors, where the terms with the least similar embeddings are ranked as top term candidates (based on the hypothesis that the embeddings of terms in domain and reference corpus will differ, while representation of non-terminological vocabulary will not differ to the same extent).

For evaluation, term candidates from the Slovenian corpus on karstology were first extracted and then manually evaluated by a domain expert. We then calculated precision at various top N values and compared the new embeddings-based extraction method to more traditional frequency-based method (comparing frequencies of words in general and domain corpora, Pollak et al. (2012)). The results are provided in Table 8. While the new method achieves slightly lower results in isolation, when combining

[21] Creating new Elmo embeddings with a lemmatized corpus was deemed to be computationally too intensive.

the two methods by adding up their individual ranks and sorting from smallest to largest, the combined method comes out on top.

top N	frequency	embeddings	combined
top 100	0.75	0.53	**0.85**
top 200	0.70	0.59	**0.74**
top 400	0.63	0.64	**0.69**
top 800	**0.58**	0.52	**0.58**

Table 8: Precision at top N of term extraction experiments on a karstology corpus.

We repeated the experiments on the ACTER dataset, which comes with annotated terms (meaning that we can evaluate recall as well). These results were less encouraging with F1 scores hovering around 0.30 which is somewhat below the state-of-the-art results (F1 score of 0.46) achieved by the winning system (Hazem et al., 2020) of the Termeval 2020 workshop (Rigouts Terryn et al., 2020). We believe one of the reasons impacting the performance was that we used the entire English Wikipedia as the general language corpus. We are currently investigating if Wikipedia may be too similar to specific terminological domains and are planning to repeat the experiments with the British National Corpus.

These experiments were not yet repeated on the news corpora. However, terminology is very closely related to keywords: while keywords usually relate to a single document, terms relate to domain words in a corpus of documents. For the tasks, where we aim to analyse and compare news corpora of, e.g., different genres or time periods, the task is closer to term extraction. These methods can therefore be considered also in T4.3 for viewpoint analysis.

6.2 Canonical term form generation

In terminology and keyword extraction, term candidates are produced as they appear in the text or at best in their lemmatized form. While this is not a significant issue for languages with little inflection, such as English, it is much more prominent in highly inflectional languages, such as Slovene and other EMBEDDIA languages.

To address this issue, we have developed a system (currently only for Slovene) for producing *canonical forms* which can be described as "word forms as they would appear in the dictionary" (e.g., . We trained the Lemmagen lemmatization algorithm (Juršič et al., 2010) on various forms in the nominative case from the Slovene morphological lexicon Sloleks (Dobrovoljc et al., 2019).

lemma	**canonical form**
strojen učenje	strojno učenje
kraški polje	kraško polje

Table 9: Examples of canonical forms

Thus we were able to recreate the canonical form of a term by canonizing each constituent word according to predefined rules on canonization. For example, for an adjective+noun phrase, if the headword noun is in the female gender, produce the female nominative form of the preceding adjective (and not the nominative male form which is the form resulting from a lemmatization). For examples of differences between lemmatized forms and cannonical forms for nouns of neutral gender, see Table 9.

6.3 Terminological semantic relations extraction

In this section, we present a study on the use of intersections of word embeddings for terminological relation extraction. The paper was accepted and presented at conference TOTh (Terminology & Onto-

logy: Theories and applications) (Grčić-Simeunović et al., 2020 (to apprear)) and will be published in upcoming conference proceedings, a preprint is available in Appendix D). This work, partly related to EMBEDDIA, presents an embeddings-based technique which can be potentially relevant also for applications in media context. For example, in keyword search the original query can be expended with embeddings for finding similar lexical items which were not listed by the query of the user, and using intersection of embeddings can be a promising technique. The method has also some potential for natural language generation in template variation, as discussed in Deliverable D2.4, where a similar study (Vintar et al. (2020)) to the one presented here was introduced.

The study is conducted on the TermFrame corpus of karstology Vintar et al. (2019). We present a method for extracting semantically related adjectives using intersections of word embeddings and a detailed manual analysis of the extracted words. The results of the first stage show high variability in precision between relations (ranging from 0.28 (for relation FUNCTION, Croatian) to 0.80 (for relation COMPOSITION, Croatian), yet for three out of five target relations the method successfully extracts numerous meaningful adjectives pertaining to the target semantic relation (FORM, COMPOSITION and CAUSE). The analysis performed in the second stage reveals several nuances of semantic similarity which we categorise into clusters. In most cases, members of a cluster share a surface linguistic component such as a suffix, prefix or word stem. Some suffixes indeed contain a semantic component pertaining to a specific relation (-genic -> CAUSE), and a shared word stem almost necessarily entails a similarity in meaning. In other cases, word embeddings allow us to retrieve synonyms with no surface similarity (podmorski – submarinski) only on the basis of their shared contexts.

This paper further analyses the approach described in Vintar et al. (2020) previously integrated in D2.4 (where we discuss how this could be useful also as a natural language generation strategy for relexicalization), while in the paper integrated here (Grčić-Simeunović et al., 2020 (to apprear)) a focus is on a more detailed discussion on linguistic analysis of semantic relations expressed by adjectives. For more details see the paper Grčić-Simeunović et al. (2020 (to apprear)) in Appendix D.

7 Term and keyword extraction for scientific literature mining

We also extended our work to extracting keywords and terms from scientific papers using word embeddings technology for supporting scientific discovery, which is a very timely application of EMBEDDIA technology given the context of COVID-19 pandemics, which was marking the society in the second period of the project. The work described in Section 7.1 utilises the EMBEDDIA keyword extractor RaKUn (presented in Deliverable D2.3) as an underlying technology in development of the system for fast search over COVID-19 papers. Next, the work in Sections 7.2 and 7.3 uses the embeddings-based technology to support new scientific discovery (the published papers are provided in Appendices C and D).

This work can be seen as part of EMBEDDIA exploitation activities, as JSI has received additional national funding for COVID-19 research and the COVID-19 explorer using the EMBEDDIA result of RaKUn keyword extractor was part of the project application.

7.1 Extending RaKUn with novel features and development of COVID-19 explorer

In the previous deliverable D2.3, we have described our novel unsupervised keyword extraction method RaKUn Škrlj et al. (2019). In summary, the main steps of RaKUn keyword extractor are as follows:

- construct a directed graph from text (vertices correspond to words, and edges to co-occurrence frequencies),

- meta vertex aggregation (aggregating vertices based on a defined similarity function),
- vertex ranking by Load centrality score, and
- multi-word keyword scoring.

The key novelty of this algorithm was the introduction of meta-vertices, i.e. vertices constructed from multiple very similar words. In the first version of the algorithm, the distances were computed as Levenshtein distances, however, we extended the distance computation to Euclidean distances between token embeddings. Here, we first compute a representation of each of the tokens, initially used to construct a token graph. Next, based on the distances between the tokens, vertices (tokens) are joined. Note that this type of vertex joining results in different token graphs, as the distances are no more of lexical, but of semantic nature. Current version of RaKUn works with FastText embeddings (pre-trained for multiple languages), and is freely available at the official repository of RaKUn.

The developed COVID-19 Explorer tool exploits RaKUn for the generation of collections of keywords for more than 50,000 COVID-19-related scientific papers. The tool is able to query the document base based on the keywords, as well as display all the documents in latent space (obtained by 2D projections of document representations obtained with doc2vec), offering a fast and intuitive explorer of COVID-19-related literature. The tool is freely available at `http://covid19explorer.ijs.si/` and in Figures 9 and 10, we present the welcome screen with user input fields and the visualisation of document embeddings, respectively.

We are currently working on a publication together with an expert from pharmaceutical science on a use case around COVID-19 Explorer. Moreover, the development of the COVID-19 Explorer is general, and can easily be adapted to other domains, including news.

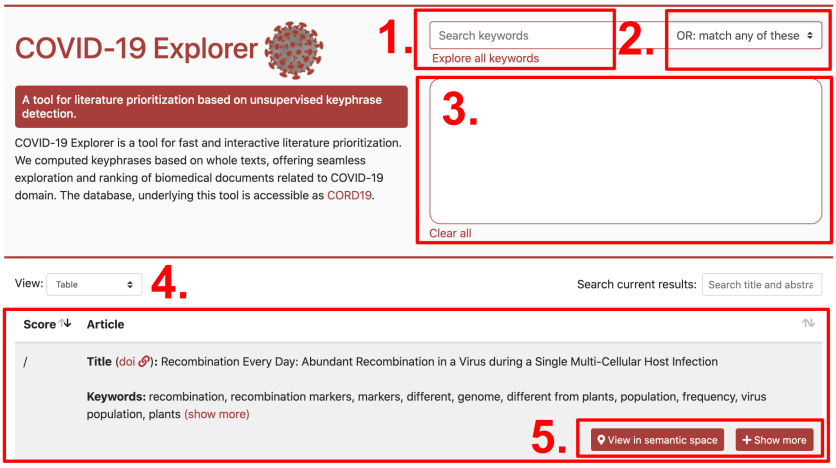


Figure 9: The COVID-19 Explorer's welcome screen and user input fields. Relevant key sections of the online tool are emphasized and numbered in red colour. Namely: 1. user keyword inspection and input field, 2. Boolean operators imposed on keywords, 3. selected keyword inspection window, 4. Dynamically updating result field displaying semantically related peer-review articles, 5. specific article detail button and 2D visualization of semantic space

7.2 Methods for term identification for novel scientific discovery using contextual embeddings

In this section, we briefly summarise our work on identifying terms for scientific discovery related to COVID-19. Again, the motivation was the period of crisis that inspired the entire scientific community to join the forces and contribute to the search of possible solutions by our knowledge and tools .

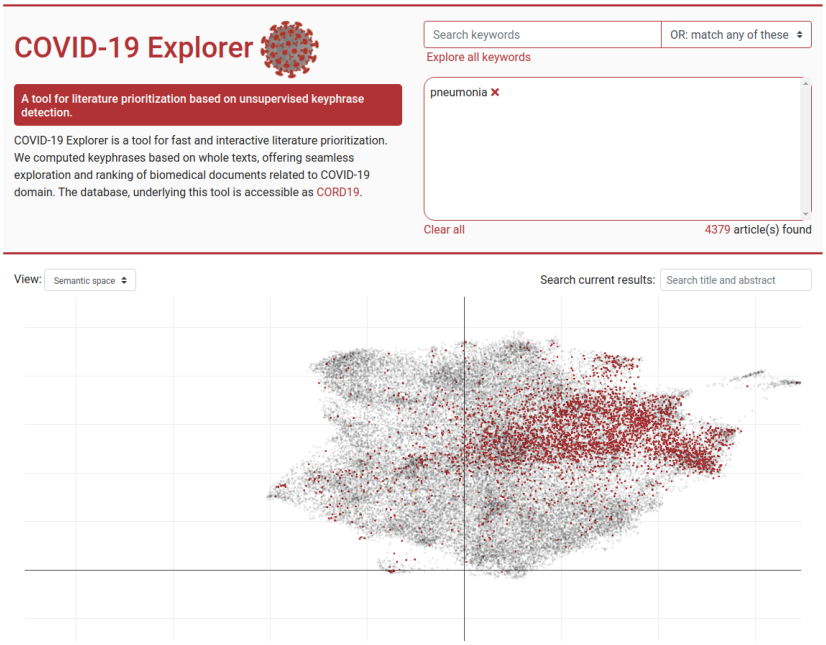


Figure 10: Visualization of the document embeddings colored by the keyword “pneumonia”.

In our paper (Martinc, Škrlj, et al., 2020), we focus on Covid-19 and identification of terms for knowledge discovery. The abundance of literature related to the widespread COVID-19 pandemic is beyond manual inspection of a single expert. Development of systems, capable of automatically processing tens of thousands of scientific publications with the aim to enrich existing empirical evidence with literature-based associations is challenging and relevant. We propose a system for contextualization of empirical expression data by approximating relations between entities, for which representations were learned from one of the largest COVID-19-related literature corpora. In order to exploit a larger scientific context by transfer learning, we propose a novel embedding generation technique that leverages SciBERT language model pretrained on a large multi-domain corpus of scientific publications and fine-tuned for domain adaptation on the CORD-19 dataset[22].

Next, we generate word representations for each word in the vocabulary, as illustrated in Figure 11. From contextual embeddings (where word representation differs for each context), we get back to static embeddings.

The main idea of our approach is to leverage semantic similarity in order to derive new scientific knowledge from an already existing one. For this to work, some initial seed concepts need to be acquired and used as a starting point. We explore two possibilities for this: seed concepts recommended by the expert and seed concepts found in the literature. Once seed concepts are acquired, we calculate their embeddings and look for semantically similar concepts by finding the concepts that are the closest to seed concepts according to the cosine distance between the embeddings. More specifically, we find a set of closest candidate concepts for each gene/protein in each seed candidate set, and the acquired candidates are ranked according to the cosine similarity. Finally, we calculate the average ranking for each candidate (i.e. by averaging ranks for each seed concept in the set) and therefore obtain the closest term candidates for each of the seed terms.

The conducted manual evaluation by the medical expert and the quantitative evaluation based on therapy targets identified in the related work suggest that the proposed method can be successfully employed for COVID-19 therapy target discovery and that it outperforms the baseline FastText static embeddings method by a large margin.

[22] https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge

This work showcases the role of terms and keywords also in the domain of scientific discovery, and could potentially be applicable for new methods for investigative journalism.

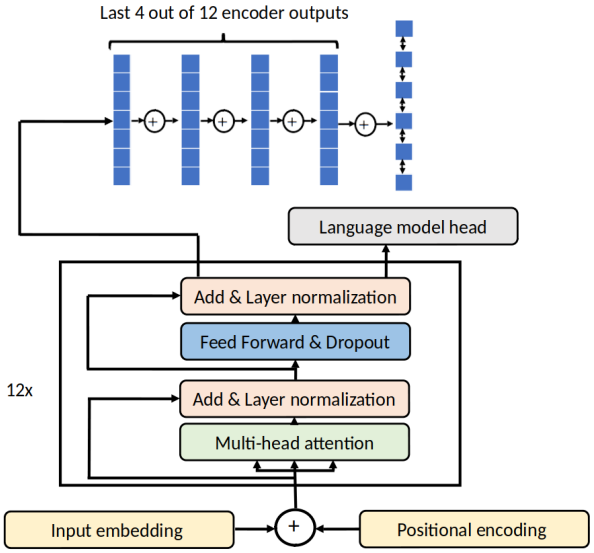


Figure 11: Extraction of word usage embeddings from BERT. Note that only the last 4 out of 12 BERT encoder layers are used for the embedding generation. This was done in accordance with the previous studies that suggested that the last four layers carry the bulk of the semantic information obtained by the model (Martinc, Kralj Novak, & Pollak, 2020).

7.3 Methods for term identification for novel bisociative scientific discovery using term analogies

Bisociative knowledge discovery is a challenging task motivated by a trend of over-specialization in research and development, which usually results in deep and relatively isolated silos of knowledge. Scientific literature too often remains closed and cited only in professional sub-communities. The information that is related across different contexts is difficult to identify using associative approaches, like the standard association rule learning (Agrawal et al., 1996) known from the data mining and machine learning literature. Therefore, the ability of literature mining methods and software tools to support the experts in their knowledge discovery processes—especially in searching for yet unexplored connections between terms of different domains—is becoming increasingly important. The field of bisociative literature-based discovery therefore aims at mining scientific literature to reveal yet uncovered connections between different fields of specialization.

In our paper (Lavrač et al., 2020a), we present the lessons learnt on bisociative knowledge discovery and extend it in our paper (Lavrač et al., 2020) (presented in Appendix B), where one of the main contributions is theoretical and experimental research on new knowledge discovery using analogies of embeddings, more specifically by analogies of terms from two different domains. This work is therefore an application of technologies developed within the project, namely embeddings and keyword extraction, to a new domain - i.e. scientific discovery.

Koestler (1964) argued that the essence of creativity lies in “perceiving of a situation or idea . . . in two self-consistent but habitually incompatible frames of reference”, and introduced the expression *bisociation* to characterize this creative act.

Starting from Koestler’s concept of bisociation, concrete bisociative patterns that are searched for in bisociative knowledge discovery include: bridging concepts, bridging graphs, and bridging by structural similarity (Kötter & Berthold, 2012):

Bridging concepts. This is the most natural type of bisociation: a concept connecting two domains. In practice, different literatures from different domains are explored, and some terms connecting the two are found. This is the kind of pattern originally explored by Swanson. These connecting terms allow us to corroborate hypotheses linking the two domains. Bridging concept in the intersection of two domains *A* and *C* is illustrated in Figure 12.

Bridging graphs. More complex bisociations are modeled by bridging graphs, in a network-representation. This is similar to bridging concepts, but in this case, what connects two different domains is a subset of *related* concepts.

Bridging by structural similarity. This is the most complex kind of bisociation, whereby, again in a network representation, subsets of concepts in each domain share structural similarities, illustrated in Figure 13.

Bisociations based on structural similarity are represented by relations and/or sub-graphs of two different, structurally-similar domains (Kötter & Berthold, 2012), illustrated in Figure 13. This type of bisociation is according to Kötter & Berthold (2012) the most abstract pattern with the potential for new cross-domain discoveries, which e.g., vertex similarity methods can identify.

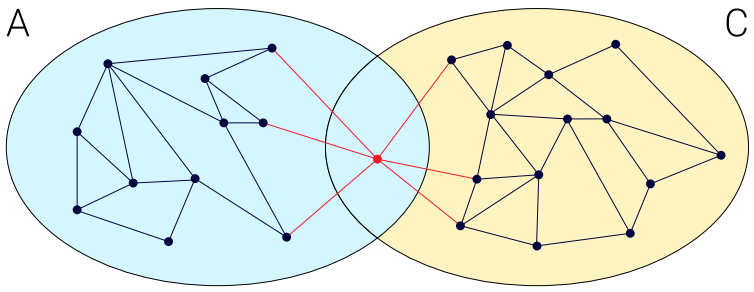


Figure 12: Bridging concept in the intersection of two literature domains *A* and *C*.

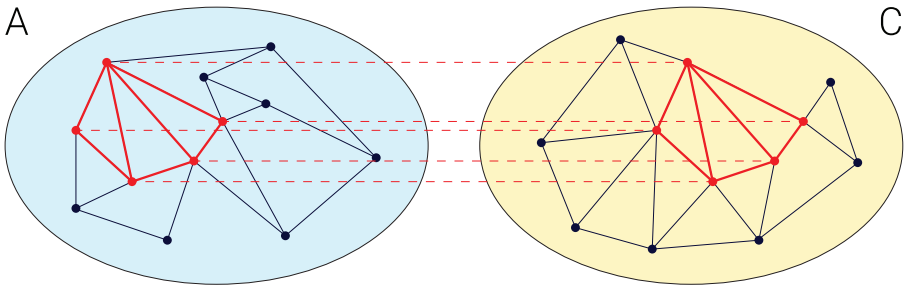


Figure 13: Bridging by graph similarity (Kötter & Berthold, 2012).

The scientific question addressed in our research is whether embeddings can be used as means for discovering *relational bisociations*, a special case of bridging by structural similarity. This new concept is illustrated in Figure 14.

Bridging by relational bisociation. We propose a particular *bridging by relational bisociation setting*, illus-

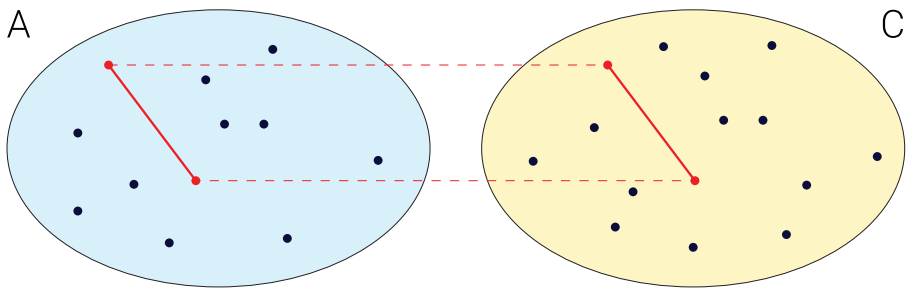


Figure 14: Concept of bridging by relational bisociation, newly introduced in our work (Lavrač et al., 2020).

trated in Figure 14, where we are interested if for a specific relation between two given concepts $a1$ and $a2$ in the first domain A one could bisociatively discover an analogous relation between concepts x and c in the second domain C, where c is a given concept (term) and x is a new concept (term) that we are trying to find. More formally, this can be written in a form of a bisociation, i.e. an analogy between two separate domains, A and C as follows:

```
a1 rel a2 == x rel c
```

In the embedding space, this analogy translates to the following equation between embeddings:

```
x = emb(a1) + emb(a2) - emb(c)
```

Finally, once x is calculated, we need to find a set of concepts from the second domain C that have an embedding representation most similar to x according to some predefined distance measure.

Methodology of bridging by relational bisociation. Proposed embedding-based bisociative literature-based discovery (LBD) methodology for creative discovery of bisociated relationships between two domains A and C consists of the following steps:

1. Select two domains A and C, i.e. two document corpora such as *circadian rhythm* and *plant defense*, respectively.
2. Train separate word embedding models for A and C to get $emb(A)$ and $emb(C)$.
3. Perform alignment of $emb(A)$ and $emb(C)$ embedding vector spaces.
4. Determine the relationships of interest in a given domain A between concepts $a1$ and $a2$ defined by the domain expert, e.g., biologist.
5. Perform the embedding-based relational LBD with a known seed concept c in C by leveraging the ability of the embedding representations to model analogy relations.
6. Evaluate a list of best ranked relational bisociations.

We present experiments conducted on the *circadian rhythm* and *plant defense* domains in collaboration with experts from biological domain. Our main goal was to identify potentially interesting new daily regulated mechanisms that are responsible for plant defence. Circadian rhythm in plants causes that some of their genes are expressed differently during the course of the day. Consequently, plants respond differently to disease-causing infection if they are infected at different times of the day (e.g., morning, noon, evening). Therefore, one of the goals of our study was to identify new gene sets that are differently expressed in different parts of the day and are important for the defense of plants against the pathogen.

After obtaining 10,494 documents from PubMed containing article titles and abstracts (4,346 from plant defence and 6,148 from circadian rhythm), we replaced gene names with synonyms gathered in previous research projects (22,265 gene names mapped into 7,863 synonyms). In addition, we pre-processed the documents to keep only gene-related terms (included in synonym list and from the gene dictionary containing additional 6,083 gene names), which resulted in a substantial reduction of the input document corpus named genesOnly dataset. On each of the two selected domains, we trained a separate FastText embeddings model (Bojanowski et al., 2017), and then aligned them into a common vector space. We opted for a supervised alignment approach, which relies on a training dictionary of identical words from both domains that are used as anchor points to learn a mapping from the source to the target space with a Procrustes alignment (Conneau et al., 2017). Next, we asked a biology expert to identify a list of genes and other terms closely related to the circadian rhythm domain and the input list of terms consisted of 15 terms.

According to our methodology, we tried to identify a list of terms (genes) related to the concept of *plant defense* in the similar way the genes from the above list are related to the concept of *circadian rhythm*. First, we calculated embedding *x* according to the following equation: `x = emb(a1) + emb(a2) - emb(c)`, where `a1` is a concept *circadian rhythm*, `a2` is a gene from the above list and `c` is a concept *plant defense*.

Finally, once `x` was calculated for each of the genes from the above list, we searched for a set of concepts from the plant defense domain that have an embeddings representation most similar to `x` according to the cosine similarity. In order to limit the results only to genes or gene related concepts, the concepts from the second domain were considered only if they appeared in the reduced genesOnly dataset. 10 genes or gene related concepts with the representation most similar to each of the calculated `x`'s were identified and given to the biology expert for the evaluation (who was given instructions to manually classify the relatedness between a candidate and the plant defense domain into the following four categories: NO, NOT AT ALL; NOT REALLY; MAYBE; YES.) Details on methodology and the results can be read in out paper Lavrač et al. (2020) in Appendix B. As one of the interesting results, the best ranked candidate obtained for the *c* term inputs CCA1 and LHY (two central genes of the circadian clock rhythm) was DMR1 (a susceptibility gene, mutation of this gene results in a higher resistance), that is a hot topic of a plant resistance research lately, showing high potential of our approach.

This research is an example of exploitation of embeddings approaches to other domains (scientific discovery), showing generality of our technologies. In addition, in future we can consider if bisociative discovery methods could be of interest for social science analysis of media text or investigative journalism.

8 Conclusions and further work

In this report we presented the work performed during the second year in the scope of Task 2.2. The main contribution is TNT-KID (Martinc, Škrlj, & Pollak, 2020), a supervised keyword extraction system. It is a novel transformer based neural tagger, which has shown a robust performance across a variety of public datasets with manually labelled keywords, as well as on EMBEDDIA media partners' datasets, with F1@10 scores between around 40% and 65% depending on the language (Croatian, Estonian, Russian and Latvian). The paper describing the method and the evaluation on public datasets is currently in revision in Natural Language Engineering journal. The deliverable also shows how a combined TNT-KID keyword extraction and TF-IDF based tagset matching have been developed in order to satisfy the need of media partners to improve the recall and get a higher number of keyword candidates. Task 2.2 is completed with this deliverable, but TNT-KID keyword extraction can further serve different tasks in WP4 and was also integrated to EMBEDDIA Media assistant (WP6). In future, we will consider a cross-lingual setting — in the scope of this task, the focus was on monolingual keyword extraction, as this was identified as a probable real application, and all involved media partners' had the training sets available. In terms of keyword related work, we have also presented the work on TEXTA Hybrid tagger, which is framed as a classification task (this work was done in collaboration between task T2.2 and

WP6, as more focus is on evaluation in industrial setting).

We have also presented advances in term extraction, a field that is closely related to keyword extraction, but more focused on extraction of domain terms from specialised document collections (and not from single documents). We report on comparison of contextual embeddings between a domain and a reference corpus, where initial results are promising. We continued our work (Repar et al., 2019) on term alignment, where our initial approach was applied to media partners' datasets, more specifically to Estonian and Russian tagset, where the goal was to match the tags between languages. The performance lagged behind the initial experiments on EU languages, but it was still satisfactory. Lower performance is due to the fact that for this language pair no large enough high quality parallel data required for dictionary induction was available. In the next step, we have investigated a possibility of using word embeddings alignment instead of dictionaries on large parallel data, and results show that in some of the settings, the precision can even be improved, while in others the results are lower than with dictionaries from parallel corpora, but with less required resources. In the reported experiments, the results were evaluated on the Eurovoc thesaurus; however, in future work, we plan to evaluate the method also in the ExM tagset alignment use case. We also report on a terminological study on adjectives, where given the semantic relation an adjective expresses, embeddings are used to find other adjectives expressing the same relation.

Finally, we presented how our unsupervised keyword extraction method RaKUn (Škrlj et al., 2019) was used to develop COVID-19 Explorer (`http://covid19explorer.ijs.si/`), a tool for fast and interactive literature prioritization. RaKUn served for computing keyphrases based on whole texts, offering seamless exploration and ranking of biomedical documents related to COVID-19 domain. The work shows usefulness of the developed keyword extraction tool beyond the media settings, and adaptation to the specific time that the project faced during the second year (COVID-19 pandemics). In similar line, we performed work partly related to EMBEDDIA, focusing on identifying terms for scientific discovery in two different settings. In the first one, the approach for bisociative knowledge discovery based on relational bisociation identification was presented and supported with experiments in biological domain (Lavrač et al., 2020) and in second, a contextual-embeddings-based method was proposed for discovering new knowledge in COVID-19 domain (Martinc, Škrlj, et al., 2020).

9 Associated outputs

The work described in this deliverable has resulted in the following resources:

Description	URL	Availability
Code for RaKUn (updated since D2.3)	`https://github.com/EMBEDDIA/RaKUn`	Public (GPL3)
Code for TNT-KID (updated since D2.3)	`https://github.com/EMBEDDIA/TNT_KID`	Public(MIT)
Code for Estonian keyword extraction	`https://gitlab.com/matej.martinc/tnt_kid_app`	Public(MIT)
Code for Latvian keyword extraction	`https://gitlab.com/boshko.koloski/tnt_kid_app_lv`	Public(MIT)
Code for Croatian keyword extraction	`https://gitlab.com/boshko.koloski/tnt_kid_app_hr`	Public(MIT)
Covid explorer	`http://covid19explorer.ijs.si/`	Public (n.a.)

Parts of this work are also described in detail in the following publications, which are attached to this deliverable as appendices:

Citation	Status	Appendix
Martinc, M., Škrlj, B., Pollak, S. (2020). TNT-KID: Transformer-based Neural Tagger for Keyword Identification. Submitted to Natural Language Engineering	Submitted	Appendix A
Lavrač, N., Martinc, M., Pollak, S., Pompe Novak, M., Cestnik, B. (2020). Bisociative Literature-Based Discovery: Lessons Learned and New Word Embedding Approach. New Generation Computing 38, 773-800.	Published	Appendix B
Martinc, M., Škrlj, B., Pirkmajer, S., Lavrač, N., Cestnik, B., Marzidovšek, M., Pollak, S. (2020). COVID-19 Therapy Target Discovery with Context-Aware Literature Mining. In Proceedings of the 23rd International Conference on Discovery Science (DS 2020), pp:109-123.	Published	Appendix C
Grčić Simeunović, L., Martinc, M., Vintar, Š. (2020). A bilingual approach to specialised adjectives through word embeddings in the karstology domain. In Proceedings of TOTH 2020.	Accepted	Appendix D

Bibliography

Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A. I., et al. (1996). Fast discovery of association rules. *Advances in Knowledge Discovery and Data Mining*, *12*(1), 307–328.

Aker, A., Paramita, M., & Gaizauskas, R. (2013). Extracting bilingual terminologies from comparable corpora. In *Proceedings of the 51st annual meeting of the association for computational linguistics (volume 1: Long papers)* (Vol. 1, pp. 402–411).

Artetxe, M., Labaka, G., & Agirre, E. (2018). A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers)* (pp. 789–798).

Baevski, A., & Auli, M. (2018). Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*.

Beliga, S., Meštrović, A., & Martinčić-Ipšić, S. (2015). An overview of graph-based keyword extraction methods and approaches. *Journal of information and organizational sciences*, *39*(1), 1–20.

Beltagy, I., Lo, K., & Cohan, A. (2019). Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Bird, S., Klein, E., & Loper, E. (2009). *Natural language processing with python* (1st ed.). O'Reilly Media, Inc.

Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, *5*, 135–146.

Bougouin, A., Boudin, F., & Daille, B. (2013). TopicRank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the international joint conference on natural language processing (ijcnlp)* (pp. 543–551).

Campos, R., Mangaravite, V., Pasquali, A., Jorge, A., Nunes, C., & Jatowt, A. (2020). Yake! keyword extraction from single documents using multiple local features. *Information Sciences*, *509*, 257–289.

Campos, R., Mangaravite, V., Pasquali, A., Jorge, A. M., Nunes, C., & Jatowt, A. (2018a). A text feature based automatic keyword extraction method for single documents. In G. Pasi, B. Piwowarski, L. Azzopardi, & A. Hanbury (Eds.), *Advances in information retrieval* (pp. 684–691). Cham: Springer International Publishing.

Campos, R., Mangaravite, V., Pasquali, A., Jorge, A. M., Nunes, C., & Jatowt, A. (2018b). Yake! collection-independent automatic keyword extractor. In *European conference on information retrieval* (pp. 806–810).

Chan, H. P., Chen, W., Wang, L., & King, I. (2019). Neural keyphrase generation via reinforcement learning with adaptive rewards. *arXiv preprint arXiv:1906.04106*.

Conneau, A., Lample, G., Ranzato, M., Denoyer, L., & Jégou, H. (2017). Word translation without parallel data. *CoRR*, *abs/1710.04087*.

Dai, Z., Yang, Z., Yang, Y., Cohen, W. W., Carbonell, J., Le, Q. V., & Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dobrovoljc, K., Krek, S., Holozan, P., Erjavec, T., Romih, M., Arhar Holdt, Š., ... Robnik-Šikonja, M. (2019). *Morphological lexicon sloleks 2.0.* Retrieved from `http://hdl.handle.net/11356/1230` (Slovenian language resource repository CLARIN.SI)

El-Beltagy, S. R., & Rafea, A. (2009). Kp-miner: A keyphrase extraction system for english and arabic documents. *Information Systems*, *34*(1), 132–144.

Forney, G. D. (1973). The viterbi algorithm. *Proceedings of the IEEE*, *61*(3), 268–278.

Gallina, Y., Boudin, F., & Daille, B. (2019). Kptimes: A large-scale dataset for keyphrase generation on news documents. *arXiv preprint arXiv:1911.12559*.

Goldberg, Y., & Orwant, J. (2013). A dataset of syntactic-ngrams over time from a very large corpus of english books. In *Second joint conference on lexical and computational semantics* (pp. 241–247).

Gollapalli, S. D., Li, X.-L., & Yang, P. (2017). Incorporating expert knowledge into keyphrase extraction. In *Thirty-first aaai conference on artificial intelligence.*

Grave, E., Joulin, A., Cissé, M., Jégou, H., et al. (2017). Efficient softmax approximation for gpus. In *Proceedings of the 34th international conference on machine learning-volume 70* (pp. 1302–1310).

Grčić-Simeunović, L., Martinc, M., & Vintar, Š. (2020 (to apprear)). A bilingual approach to specialised adjectives through word embeddings in the karstology domain..

Gu, J., Lu, Z., Li, H., & Li, V. O. (2016). Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Hasan, K. S., & Ng, V. (2014). Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 1: Long papers)* (Vol. 1, pp. 1262–1273).

Hazem, A., Bouhandi, M., Boudin, F., & Daille, B. (2020). Termeval 2020: Taln-ls2n system for automatic term extraction. In *Proceedings of the 6th international workshop on computational terminology* (pp. 95–100).

Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on empirical methods in natural language processing* (pp. 216–223).

Jain, S., & Wallace, B. C. (2019). Attention is not explanation. *arXiv preprint arXiv:1902.10186*.

Jawahar, G., Sagot, B., & Seddah, D. (2019). What does bert learn about the structure of language?.

Juršič, M., Mozetič, I., Erjavec, T., & Lavrač, N. (2010). Lemmagen: Multilingual lemmatisation with induced ripple-down rules. *Journal of Universal Computer Science*, *16*(9), 1190–1214.

Kim, S. N., Medelyan, O., Kan, M.-Y., & Baldwin, T. (2010). Semeval-2010 task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th international workshop on semantic evaluation* (pp. 21–26). Stroudsburg, PA, USA: Association for Computational Linguistics.

Koestler, A. (1964). *The act of creation.* Hutchinson.

Kötter, T., & Berthold, M. (2012). From information networks to bisociative information networks. In *Bisociative knowledge discovery* (p. 33-–50). Springer.

Krapivin, M., Autaeu, A., & Marchese, M. (2009). *Large dataset for keyphrases extraction* (Tech. Rep.). University of Trento.

Krek, S., Holdt, Š. A., Erjavec, T., Čibej, J., Repar, A., Gantar, P., ... Dobrovoljc, K. (2020). Gigafida 2.0: The reference corpus of written standard slovene. In *Proceedings of the 12th language resources and evaluation conference* (pp. 3340–3345).

Kudo, T., & Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Lafferty, J., McCallum, A., & Pereira, F. C. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Lavrač, N., Martinc, M., Pollak, S., Novak, M. P., & Cestnik, B. (2020, October). Bisociative literature-based discovery: Lessons learned and new word embedding approach. *New Generation Computing*, *38*(4), 773–800. Retrieved from `https://doi.org/10.1007/s00354-020-00108-w` doi: 10.1007/s00354-020-00108-w

Lavrač, N., Martinc, M., Pollak, S., & Cestnik, B. (2020a). Bisociative literature-based discovery: Lessons learned and new prospects. In *Proceedings of the 11th international conference on computational creativity (iccc'20)* (p. 139-145).

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Loper, E., & Bird, S. (2002). Nltk: The natural language toolkit. In *Proceedings of the acl-02 workshop on effective tools and methodologies for teaching natural language processing and computational linguistics.*

Luan, Y., Ostendorf, M., & Hajishirzi, H. (2017). Scientific information extraction with semi-supervised neural tagging. *arXiv preprint arXiv:1708.06075*.

Martinc, M., Kralj Novak, P., & Pollak, S. (2020). Leveraging contextual embeddings for detecting diachronic semantic shift. In *Proceedings of the 12th language resources and evaluation conference* (pp. 4811–4819). Marseille, France: European Language Resources Association. Retrieved from `https://www.aclweb.org/anthology/2020.lrec-1.592`

Martinc, M., Škrlj, B., Pirkmajer, S., Lavrač, N., Cestnik, B., Marzidovšek, M., & Pollak, S. (2020). COVID-19 therapy target discovery with context-aware literature mining. *Discovery Science. DS 2020. Lecture Notes in Computer Science*. doi: https://doi.org/10.1007/978-3-030-61527-7_8

Martinc, M., Škrlj, B., & Pollak, S. (2020). TNT-KID: Transformer-based neural tagger for keyword identification. *arXiv preprint: arXiv:2003.09166*.

Medelyan, O., Frank, E., & Witten, I. H. (2009). Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 conference on empirical methods in natural language processing: Volume 3-volume 3* (pp. 1318–1327).

Meng, R., Yuan, X., Wang, T., Brusilovsky, P., Trischler, A., & He, D. (2019). Does order matter? an empirical study on generating multiple keyphrases as a sequence. *arXiv preprint arXiv:1909.03590*.

Meng, R., Zhao, S., Han, S., He, D., Brusilovsky, P., & Chi, Y. (2017). Deep keyphrase generation. *arXiv preprint arXiv:1704.06879*.

Mihalcea, R., & Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 404–411).

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Nguyen, T. D., & Kan, M.-Y. (2007). Keyphrase extraction in scientific publications. In *International conference on asian digital libraries* (pp. 317–326).

Orasmaa, S., Petmanson, T., Tkachenko, A., Laur, S., & Kaalep, H.-J. (2016, may). Estnltk - nlp toolkit for estonian. In N. C. C. Chair) et al. (Eds.), *Proceedings of the tenth international conference on language resources and evaluation (lrec 2016).* Paris, France: European Language Resources Association (ELRA).

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... others (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems* (pp. 8024–8035).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018a). Deep contextualized word representations. In *Proc. of naacl.*

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018b). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Pollak, S., Repar, A., Martinc, M., & Podpečan, V. (2019). Karst exploration: Extracting terms and definitions from Karst domain corpus. In *Electronic lexicography in the 21st century: proceedings of eLex 2019 conference.*

Pollak, S., Vavpetič, A., Kranjc, J., Lavrač, N., & Vintar, v. (2012). NLP workflow for on-line definition extraction from English and Slovene text corpora. In *11th conference on natural language processing, KONVENS 2012 - empirical methods in natural language processing, vienna, austria* (pp. 53–60).

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, *1*(8).

Repar, A., Martinc, M., & Pollak, S. (2019, Nov). Reproduction, replication, analysis and adaptation of a term alignment approach. *Language Resources and Evaluation*. Retrieved from `https://doi.org/10.1007/s10579-019-09477-1` doi: 10.1007/s10579-019-09477-1

Rigouts Terryn, A., Hoste, V., Drouin, P., & Lefever, E. (2020). Termeval 2020: Shared task on automatic term extraction using the annotated corpora for term extraction research (acter) dataset. In *6th international workshop on computational terminology (computerm 2020)* (pp. 85–94).

Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1–20.

Sahrawat, D., Mahata, D., Kulkarni, M., Zhang, H., Gosangi, R., Stent, A., ... Zimmermann, R. (2020). Keyphrase extraction from scholarly articles as sequence labeling using contextualized embeddings. In *Proceedings of european conference on information retrieval (ecir 2020)* (pp. 328–335).

Škrlj, B., Repar, A., & Pollak, S. (2019). RaKUn: Rank-based keyword extraction via unsupervised learning and meta vertex aggregation. In *International conference on statistical language and speech processing* (pp. 311–323).

Steinberger, R., Pouliquen, B., & Hagman, J. (2002). Cross-lingual document similarity calculation using the multilingual thesaurus eurovoc. *Computational Linguistics and Intelligent Text Processing*, 101–121.

Sterckx, L., Demeester, T., Deleu, J., & Develder, C. (2015). Topical word importance for fast keyphrase extraction. In *Proceedings of the 24th international conference on world wide web* (pp. 121–122).

Terryn, A. R., Hoste, V., & Lefever, E. (2019). In no uncertain terms: a dataset for monolingual and multilingual automatic term extraction from comparable corpora. *Language Resources and Evaluation*, 1–34.

Vaik, K., Asula, M., & Sirel, R. (2020, May). Hybrid tagger - an industry-driven solution for extreme multi-label text classification. In *Proceedings of the lrec2020 industry track* (pp. 26–30). Marseille, France: European Language Resources Association (ELRA). Retrieved from `https://www.aclweb.org/anthology/2020.lrec2020industrytrack-1.6`

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).

Vintar, Š. (2010). Bilingual term recognition revisited: The bag-of-equivalents term alignment approach and its evaluation. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, *16*(2), 141–158.

Vintar, Š., Saksida, A., Vrtovec, K., & Stepišnik, U. (2019). Modelling specialized knowledge with conceptual frames: The TermFrame approach to a structured visual domain representation. In *Proceedings of elex* (pp. 305–318).

Vintar, Š., Simeunović, L. G., Martinc, M., Pollak, S., & Stepisnik, U. (2020). Mining semantic relations from comparable corpora through intersections of word embeddings. In *Proceedings of the 13th workshop on building and using comparable corpora, bucc@lrec 2020* (pp. 29–34).

Wan, X., & Xiao, J. (2008). Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the aaai conference* (Vol. 8, pp. 855–860).

Witten, I. H., Paynter, G. W., Frank, E., Gutwin, C., & Nevill-Manning, C. G. (2005). Kea: Practical automated keyphrase extraction. In *Design and usability of digital libraries: Case studies in the asia pacific* (pp. 129–152). IGI Global.

Yuan, X., Wang, T., Meng, R., Thaker, K., Brusilovsky, P., He, D., & Trischler, A. (2019). One size does not fit all: Generating and evaluating variable number of keyphrases. *arXiv preprint arXiv:1810.05241*.

Appendix A: TNT-KID: Transformer-based Neural Tagger for Keyword Identification

ARTICLE

TNT-KID: Transformer-based Neural Tagger for Keyword Identification

Matej Martinc*[1,2], Blaž Škrlj[1,2], Senja Pollak[1]

[1]Jožef Stefan Institute,
Department of Knowledge Technologies, Jamova 39, 1000 Ljubljana, Slovenia
[2]Jožef Stefan International Postgraduate School,
Department of Knowledge Technologies, Jamova 39, 1000 Ljubljana, Slovenia
*Corresponding author. Email:matej.martinc@ijs.si

(Received xx xxx xxx; revised xx xxx xxx; accepted xx xxx xxx)

Abstract
With growing amounts of available textual data, development of algorithms capable of automatic analysis, categorization and summarization of these data has become a necessity. In this research we present a novel algorithm for keyword identification, i.e., an extraction of one or multi-word phrases representing key aspects of a given document, called Transformer-based Neural Tagger for Keyword IDentification (TNT-KID). By adapting the transformer architecture for a specific task at hand and leveraging language model pretraining on a domain specific corpus, the model is capable of overcoming deficiencies of both supervised and unsupervised state-of-the-art approaches to keyword extraction by offering competitive and robust performance on a variety of different datasets while requiring only a fraction of manually labeled data required by the best performing systems. This study also offers thorough error analysis with valuable insights into the inner workings of the model and an ablation study measuring the influence of specific components of the keyword identification workflow on the overall performance.

1. Introduction

With the exponential growth in amount of available textual resources, organization, categorization and summarization of these data presents a challenge, the extent of which becomes even more apparent when it is taken into the account that a majority of these resources do not contain any adequate meta information. Manual categorization and tagging of documents is unfeasible due to a large amount of data, therefore development of algorithms capable of tackling these tasks automatically and efficiently has become a necessity (Firoozeh *et al.* 2020).

One of the crucial tasks for organization of textual resources is keyword identification, which deals with automatic extraction of words that represent crucial semantic aspects of the text and summarize its content. First automated solutions to keyword extraction have been proposed more than a decade ago (Witten *et al.* 2005; Mihalcea & Tarau 2004) and the task is currently again gaining traction, with several new algorithms proposed in the recent years. Novel unsupervised approaches, such as RaKUn (Škrlj *et al.* 2019) and YAKE (Campos *et al.* 2018b), work fairly well and have some advantages over supervised approaches, as they are language and genre independent, do not require any training and are computationally undemanding. On the other hand, they also have a couple of crucial deficiencies:

- Term frequency - inverse document frequency (TfIdf) and graph based features, such as PageRank, used by these systems to detect the importance of each word in the document, are based only on simple statistics like word occurrence and co-occurrence, and are therefore unable to grasp the entire semantic information of the text.
- Since these systems cannot be trained, they can not be adapted to the specifics of the syntax, semantics, content, genre and keyword assignment regime of a specific text (e.g., a variance in a number of keywords).

These deficiencies result in a much worse performance when compared to the state-of-the-art supervised algorithms (see Table 2), which have a direct access to the gold standard keyword set for each text during the training phase, enabling more efficient adaptation. The newest supervised neural algorithms (Meng *et al.* 2019; Yuan *et al.* 2019) therefore achieve excellent performance under satisfactory training conditions and can model semantic relations much more efficiently than algorithms based on simpler word frequency statistics. On the other hand, these algorithms are resource demanding, require vast amount of domain specific data for training and can therefore not be used in domains and languages that lack manually labeled resources of sufficient size.

In this research we propose Transformer-based Neural Tagger for Keyword IDentification (TNT-KID)[a] that is capable of overcoming the aforementioned deficiencies of supervised and unsupervised approaches. We show that, while requiring only a fraction of manually labeled data required by other neural approaches, the proposed approach achieves performance comparable to the state-of-the-art supervised approaches on test sets for which a lot of manually labeled training data is available. On the other hand, if training data that is sufficiently similar to the test data is scarce, our model outperforms state-of-the-art approaches by a large margin. This is achieved by leveraging the transfer learning technique, where a keyword tagger is first trained in an unsupervised way as a language model on a large corpus and then fine-tuned on a (usually) small-sized corpus with manually labeled keywords. By conducting experiments on two different domains, computer science articles and news, we show that the language model pretraining allows the algorithm to successfully adapt to a specific domain and grasp the semantic information of the text, which drastically reduces the needed amount of labeled data for training the keyword detector.

The transfer learning technique (Peters *et al.* 2018a; Howard & Ruder 2018), which has recently become a well established procedure in the field of natural language processing (NLP), in a large majority of cases relies on very large unlabeled textual resources used for language model pretraining. For example, a well known English BERT model (Devlin *et al.* 2018) was pretrained on the Google Books Corpus (Goldberg & Orwant 2013) (800 million tokens) and Wikipedia (2,500 million tokens). On the other hand, we show that smaller unlabeled domain specific corpora (87 million tokens for computer science and 232 million tokens for news domain) can be successfully used for unsupervised pretraining, which makes the proposed approach easily transferable to languages with less textual resources and also makes training more feasible in terms of time and computer resources available.

Unlike most other proposed state-of-the-art neural keyword extractors (Meng *et al.* 2017 2019; Yuan *et al.* 2019), we do not employ recurrent neural networks but instead opt for a transformer architecture (Vaswani *et al.* 2017), which has not been widely employed for the task at hand. In fact, the study by Sahrawat *et al.* (2020) is the only study we are aware of that employs transformers for the keyword extraction task. Another difference between our approach and most very recent state-of-the-art approaches from the related work is also task formulation. While Meng *et al.* (2017 2019) and Yuan *et al.* (2019) formulate a keyword extraction task as a sequence-to-sequence generation task, where the classifier is trained to generate an output sequence of keyword tokens step by step according to the input sequence and the previous generated output tokens, we formulate

[a]Code is available under the MIT license at `https://gitlab.com/matej.martinc/tnt_kid/`.

a keyword extraction task as a sequence labeling task, similar as in Gollapalli *et al.* (2017), Luan *et al.* (2017) and Sahrawat *et al.* (2020).

Besides presenting a novel keyword extraction procedure, the study also offers an extensive error analysis, in which the visualization of transformer attention heads is used to gain insights into inner workings of the model and in which we pinpoint key factors responsible for the differences in performance of TNT-KID and other state-of-the-art approaches. Finally, this study also offers a systematic evaluation of several building blocks and techniques used in a keyword extraction workflow in a form of an ablation study. Besides determining the extent to which transfer learning affects the performance of the keyword extractor, we also compare two different pretraining objectives, autoregressive language modelling and masked language modelling (Devlin *et al.* 2018), and measure the influence of transformer architecture adaptations, a choice of input encoding scheme and the addition of part-of-speech (POS) tags information on the performance of the model.

The paper is structured as follows. Section 2 addresses the related work on keyword identification and covers several supervised and unsupervised approaches to the task at hand. Section 3 describes the methodology of our approach, while in Section 4 we present the datasets, conducted experiments and results. Section 5 covers error analysis, Section 6 presents the conducted ablation study, while the conclusions and directions for further work are addressed in Section 7.

2. Related work

This section overviews selected methods for keyword extraction, supervised in Section 2.1 and unsupervised in Section 2.2. The related work is somewhat focused on the newest keyword extraction methods, therefore for a more comprehensive survey of slightly older methods, we refer the reader to Hasan & Ng (2014).

2.1 Supervised keyword extraction methods

Traditional supervised approaches to keyword extraction considered the task as a two step process (the same is true for unsupervised approaches). First, a number of syntactic and lexical features are used to extract keyword candidates from the text. Secondly, the extracted candidates are ranked according to different heuristics and the top n candidates are selected as keywords (Yuan *et al.* 2019). One of the first supervised approaches to keyword extraction was proposed by Witten *et al.* (2005), whose algorithm named KEA uses only TfIdf and the term's position in the text as features for term identification. These features are fed to the Naive Bayes classifier, which is used to determine for each word or phrase in the text if it is a keyword or not. Medelyan *et al.* (2009) managed to build on the KEA approach and proposed the *Maui* algorithm, which also relies on the Naive Bayes classifier for candidate selection but employs additional semantic features, such as e.g., *node degree*, which quantifies the semantic relatedness of a candidate to other candidates, and *Wikipedia-based keyphraseness*, which is the likelihood of a phrase being a link in the Wikipedia.

A more recent supervised approach is a so-called sequence labelling approach to keyword extraction by Gollapalli *et al.* (2017), where the idea is to train a keyword tagger using token-based linguistic, syntactic and structural features. The approach relies on a trained Conditional Random Field (CRF) tagger and the authors demonstrated that this approach is capable of working on-par with slightly older state-of-the-art systems that rely on information from the Wikipedia and citation networks, even if only within-document features are used. Another sequence labeling approach proposed by Luan *et al.* (2017) builds a sophisticated neural network by combing an input layer comprising a concatenation of word, character and part-of-speech embeddings, a bidirectional Long Short-Term Memory (BiLSTM) layer and a CRF tagging layer. They also propose a new semi-supervised graph based training regime for training the network.

4 Natural Language Engineering

Some of the most recent state-of-the-art approaches to keyword detection consider the problem as a sequence-to-sequence generation task. The first research leveraging this tactic was proposed by Meng *et al.* (2017), employing a generative model for keyword prediction with a recurrent encoder-decoder framework with an attention mechanism capable of detecting keywords in the input text sequence and also potentially finding keywords that do not appear in the text. Since finding absent keywords involves a very hard problem of finding a correct class in a set of usually thousands of unbalanced classes, their model also employs a copying mechanism (Gu *et al.* 2016) based on positional information, in order to allow the model to find important keywords present in the text, which is a much easier problem.

Very recently, the model proposed by Meng *et al.* (2017) has been somewhat improved by investigating different ways in which the target keywords can be fed to a classifier during the training phase. While the original system used a so-called *one-to-one* approach, where a training example consists of an input text and a single keyword, the improved model (Meng *et al.* 2019) now employs a *one-to-seq* approach, where an input text is matched with a concatenated sequence made of all the keywords for a specific text. The study also shows that the order of the keywords in the text matters. The best performing model from Meng *et al.* (2019), named CopyRNN, is used in our experiments for the comparison with the state-of-the-art (see Section 4). A *one-to-seq* approach has been even further improved by Yuan *et al.* (2019), who incorporated two diversity mechanisms into the model. The mechanisms (called *semantic coverage* and *orthogonal regularization*) constrain the over-all inner representation of a generated keyword sequence to be semantically similar to the overall meaning of the source text and therefore force the model to produce diverse keywords. The resulting model leveraging these mechanisms has been named CatSeqD and is also used in our experiments for the comparison between TNT-KID and the state-of-the-art.

A further improvement of the generative approach towards keyword detection has been proposed by Chan *et al.* (2019), who integrated a reinforcement learning (RL) objective into the keyphrase generation approach proposed by Yuan *et al.* (2019). This is done by introducing an adaptive reward function that encourages the model to generate sufficient amount of accurate keyphrases. They also propose a new Wikipedia based evaluation method that can more robustly evaluate the quality of the predicted keyphrases by also considering name variations of the ground-truth keyphrases.

We are aware of one study that tackled keyword detection with transformers. Sahrawat *et al.* (2020) fed contextual embeddings generated using several transformer and recurrent architectures (BERT (Devlin *et al.* 2018), RoBERTa (Liu *et al.* 2019), GPT-2 (Radford *et al.* 2019), ELMo (Peters *et al.* 2018b), etc.) into two distinct neural architectures, a bidirectional Long short-term memory network (BiLSTM) and a BiLSTM network with an additional Conditional random fields layer (BiLSTM-CRF). Same as in Gollapalli *et al.* (2017), they formulate a keyword extraction task as a sequence labelling approach, in which each word in the document is assigned one of the three possible labels: k_b denotes that the word is the first word in a keyphrase, k_i means that the word is inside a keyphrase, and k_o indicates that the word is not part of a keyphrase.

The study shows that contextual embeddings generated by transformer architectures generally perform better than static (e.g., FastText embeddings (Bojanowski *et al.* 2017)) and among them BERT showcases the best performance. Since all of the keyword detection experiments are conducted on scientific articles, they also test SciBERT (Beltagy *et al.* 2019), a version of BERT pretrained on a large multi-domain corpus of scientific publications containing 1.14M papers sampled from Semantic Scholar. They observe that this genre specific pretraining on texts of the same genre as the texts in the keyword datasets, slightly improves the performance of the model. They also report significant gains in performance when the BiLSTM-CRF architecture is used instead of BiLSTM.

The neural sequence-to-sequence models are capable of outperforming all older supervised and unsupervised models by a large margin but do require a very large training corpora with tens of thousands of documents for successful training. This means that their use is limited only to

languages (and genres) in which large corpora with manually labeled keywords exist. On the other hand, the study by Sahrawat *et al.* (2020) indicates that the employment of contextual embeddings reduces the need for a large dataset with manually labeled keywords. These models can therefore be deployed directly on smaller datasets by leveraging semantic information already encoded in contextual embeddings.

2.2 Unsupervised keyword extraction methods

The previous section discussed recently emerged methods for keyword extraction that operate in a supervised learning setting and can be data-intensive and time consuming. Unsupervised keyword detectors can tackle these two problems, yet at the cost of the reduced overall performance.

Unsupervised approaches need no training and can be applied directly without relying on a gold standard document collection. They can be divided into statistical and graph-based methods:

- Statistical methods, such as KP-MINER (El-Beltagy & Rafea 2009), RAKE (Rose *et al.* 2010) and YAKE (Campos *et al.* 2018ab), use statistical characteristics of the texts to capture keywords.
- Graph-based methods, such as TextRank (Mihalcea & Tarau 2004), Single Rank (Wan & Xiao 2008), TopicRank (Bougouin *et al.* 2013), Topical PageRank (Sterckx *et al.* 2015) and RaKUn (Škrlj *et al.* 2019) build graphs to rank words based on their position in the graph.

Among the statistical approaches, the state-of-the-art keyword extraction algorithm is YAKE (Campos *et al.* 2018ab). It defines a set of features capturing keyword characteristics which are heuristically combined to assign a single score to every keyword. These features include casing, position, frequency, relatedness to context and dispersion of a specific term.

One of the first graph-based methods for keyword detection is TextRank (Mihalcea & Tarau 2004), which first extracts a lexical graph from text documents and then leverages Google's PageRank algorithm to rank vertices in the graph according to their importance inside a graph. This approach was somewhat upgraded by TopicRank (Bougouin *et al.* 2013), where candidate keywords are additionally clustered into topics and used as vertices in the graph. Keywords are detected by selecting a candidate from each of the top-ranked topics. The most recent graph-based keyword detector is RaKUn (Škrlj *et al.* 2019) that employs several new techniques for graph construction and vertice ranking. First, initial lexical graph is expanded and adapted with the introduction of meta-vertices, i.e., aggregates of existing vertices. Second, for keyword detection and ranking, a graph-theoretic *load centrality* measure is used along with the implemented graph redundancy filters.

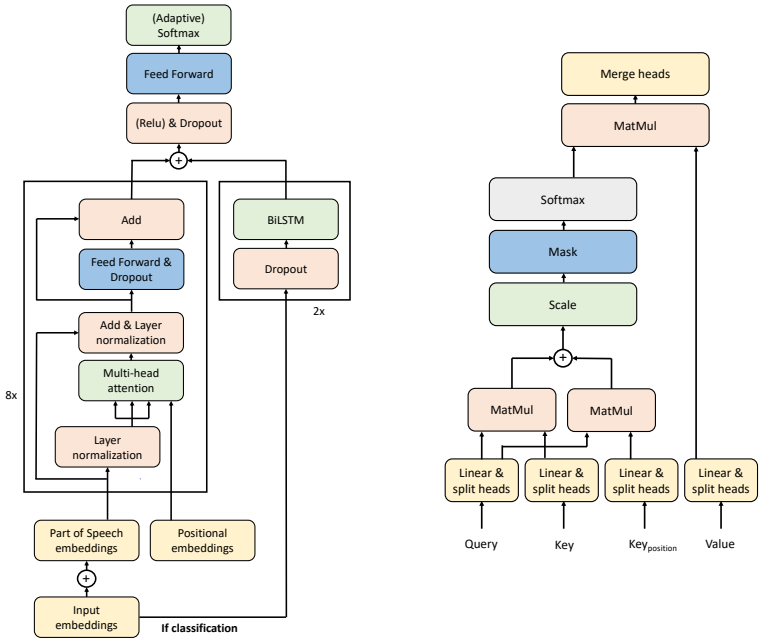
3. Methodology

This section presents the methodology of our approach. Section 3.1 presents the architecture of the neural model, Section 3.2 covers the transfer learning techniques used, Section 3.3 explains how the final fine-tuning phase of the keyword detection workflow is conducted and Section 3.4 covers evaluation of the model.

3.1 Architecture

The model follows an architectural design of an original transformer encoder (Vaswani *et al.* 2017) and is presented in Figure 1a. Same as in the GPT-2 architecture (Radford *et al.* 2019), the encoder consists of a normalization layer that is followed by a multi-head attention mechanism. A residual connection is employed around the attention mechanism, which is followed by another

6 Natural Language Engineering



(a) Model architecture. (b) The attention mechanism.

Figure 1: TNT-KID's architecture overview.

layer normalization. This is followed by the fully connected feed-forward and dropout layers, around which another residual connection is employed.

For two distinct training phases, language model pretraining and fine-tuning, two distinct "heads" are added on top of the encoder, which is identical for both phases and therefore allows for the transfer of weights from the pretraining phase to the fine-tuning phase. The language model head predicts the probability for each word in the vocabulary that it appears at a specific position in the sequence and consists of a dropout layer and a feed forward layer of size $\mathrm{SL} * |V|$, where SL stands for sequence length (i.e., a number of words in the input text) and $|V|$ stands for the vocabulary size. This is followed by the adaptive softmax layer (Grave *et al.* 2017) (see description below).

During fine-tuning, the language model head is replaced with a token classification head, in which we apply ReLu non-linearity and dropout to the encoder output, and then feed the output to the feed forward classification layer of size $\mathrm{SL} * \mathrm{NC}$, where NC stands for the number of classes (in our case 2, since we model keyword extraction as a binary classification task, see Section 3.3 for more details). Finally, a softmax layer is added in order to obtain probabilities for each class.

We also propose some significant modifications of the original GPT-2 architecture. First, we propose a re-parametrization of the attention mechanism that allows to model the relation between a token and its position (see Figure 1b). Note that standard scaled dot-product attention (Vaswani *et al.* 2017) requires three inputs, a so-called *query, key, value* matrix representations of the embedded input sequence and its positional information (i.e., element wise addition of input embeddings and positional embeddings) and the idea is to obtain attention scores (in a shape of an attention matrix) for each relation between tokens inside these inputs by first multiplying *query* (Q) and transposed *key* (K) matrix representations, applying scaling and softmax functions, and finally multiplying the resulting normalized matrix with the *value* (V) matrix, or more formally,

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where d_k represents the scaling factor, usually corresponding to the first dimension of the *key* matrix. On the other hand, we propose to add an additional positional input representation matrix K_{position} and model attention with the following equation:

$$\text{Attention}(Q, K, V, K_{pos}) = \text{softmax}\left(\frac{QK^T + QK^T_{position}}{\sqrt{d_k}}\right)V$$

The reason behind this modification is connected with the hypothesis, that token position is important in the keyword identification task and with this re-parametrization the model is capable of directly modelling the importance of relation between each token and each position. Note that we use relative positional embeddings for representing the positional information, same as in Dai *et al.* (2019), where the main idea is to only encode the relative positional information in the hidden states instead of the absolute.

Second, besides the text input, we also experiment with the additional part-of-speech (POS) tag sequence as an input. This sequence is first embedded and then added to the word embedding matrix. Note that this additional input is optional and is not included in the model for which the results are presented in Section 4.3 due to marginal effect on the performance of the model in the proposed experimental setting (see Section 6).

While the modifications presented above affect both training phases (i.e., the language model pretraining and the token classification fine-tuning), the third modification only affects the language model pretraining (see Section 3.2) and involves replacing the standard input embedding layer and softmax function with adaptive input representations (Baevski & Auli 2018) and an adaptive softmax (Grave *et al.* 2017). The main idea is to exploit the unbalanced word distribution to form word clusters containing words with similar appearance probabilities. The entire vocabulary is split into a smaller cluster containing about 10 percent of words that appear most frequently, a second slightly bigger cluster that contains words that appear less frequently and a third cluster that contains all the other words that appear rarely in the corpus. During language model training, instead of predicting an entire vocabulary distribution at each time step, the model first tries to predict a cluster in which a target word appears in and after that predicts a vocabulary distribution just for the words in that cluster. Since in a large majority of cases the target word belongs to the smallest cluster containing most frequent words, the model in most cases only needs to generate probability distribution for a tenth of a vocabulary, which drastically reduces the memory requirements and time complexity of the model at the expense of a marginal drop in performance.

We also present the modification, which only affects the fine-tuning token classification phase (see Section 3.3). During this phase, a two layer randomly initialised encoder, consisting of dropout and two bidirectional Long short-term memory (BiLSTM) layers, is added (with element-wise summation) to the output of the transformer encoder. The initial motivation behind this adaptation is connected with findings from the related work which suggest that recurrent layers are quite successful at modelling positional importance of tokens in the keyword detection task (Meng *et al.* 2017; Yuan *et al.* 2019) and by the study of Sahrawat *et al.* (2020), who also reported good results when a BiLSTM classifier and contextual embeddings generated by transformer architectures were employed for keyword detection. Also, the results of the initial experiments suggested that some performance gains can in fact be achieved by employing this modification.

In terms of computational complexity, a self-attention layer complexity is $O(n^2 * d)$ and the complexity of the recurrent layer is $O(n * d^2)$, where n is the sequence length and d is the embedding size (Vaswani *et al.* 2017). The standard TNT-KID model employs sequence size of 256, embedding size of 512 and 8 attention layers. The complexity of the model without

recurrent encoder is therefore $256^2 * 512 * 8 = 268435456$. By adding the recurrent encoder with two recurrent bidirectional layers (which is the same as adding 4 recurrent layers, since each bidirectional layer contains two unidirectional LSTM layers), the complexity increases by $256 * 512^2 * 4 = 268435456$, which means that the model with the additional recurrent encoder conducts token classification roughly two times slower than the model without the encoder. Note that this addition does not affect the language model pretraining, which tends to be the more time demanding task due to larger corpora involved.

Finally, we also experiment with an employment of the BiLSTM-CRF classification head on top of the transformer encoder, in order to compare our proposed approach to the approach proposed by Sahrawat *et al.* (2020) (see Section 6 for more details about the results of this experiment). For this experiment, during the fine-tuning token classification phase, the token classification head described above is replaced with a BiLSTM-CRF classification head proposed by Sahrawat *et al.* (2020), containing one BiLSTM layer and a CRF (Lafferty *et al.* 2001) layer.[b] Outputs of the BiLSTM $f = f_1, ..., f_n$ are fed as inputs to a CRF layer, which returns the output score $s(f, y)$ for each possible label sequence according to the following equation:

$$s(f, y) = \sum_{t=1}^{n} \tau_{y_{t-1}, y_t} + f_{t, y_t}$$

τ_{y_{t-1}, y_t} is a transition matrix representing the transition score from class y_{t-1} to y_t. The final probability of each label sequence score is generated by exponentiating the scores and normalizing over all possible output label sequences:

$$p(y|f) = \frac{exp(s(f, y))}{\sum_{y'} exp(s(f', y'))}$$

To find the optimal sequence of labels efficiently, the CRF layer uses the Viterbi algorithm (Forney 1973).

3.2 Transfer learning

Our approach relies on a transfer learning technique (Howard & Ruder 2018; Devlin *et al.* 2018), where a neural model is first pretrained as a language model on a large corpus. This model is then fine-tuned for each specific keyword detection task on each specific manually labeled corpus by adding and training the token classification head described in the previous section. With this approach, the syntactic and semantic knowledge of the pretrained language model is transferred and leveraged in the keyword detection task, improving the detection on datasets that are too small for the successful semantic and syntactic generalization of the neural model.

In the transfer learning scenario, two distinct pretraining objectives can be considered. First is the autoregressive language modelling where the task can be formally defined as predicting a probability distribution of words from the fixed size vocabulary V, for word w_t, given the historical sequence $w_{1:t-1} = [w_1, ..., w_{t-1}]$. This pretraining regime was used in the GPT-2 model (Radford *et al.* 2019) that we modified. Since in the standard transformer architecture self-attention is applied to an entire surrounding context of a specific word (i.e., the words that appear after a specific word in each input sequence are also used in the self-attention calculation), we employ obfuscation masking to the right context of each word when the autoregressive language model objective is used, in order to restrict the information only to the prior words in the sentence (plus the word itself) and prevent target leakage (see Radford *et al.* (2019) for details on the masking procedure).

[b] Note that in the experiments in which we employ BiLSTM-CRF, we do not add an additional two layer BiLSTM encoder described above to the output of the transformer encoder.

Another option is a masked language modelling objective, first proposed by Devlin *et al.* (2018). Here, a percentage of words from the input sequence is masked in advance, and the objective is to predict these masked words from an unmasked context. This allows the model to leverage both left and right context, or more formally, the token w_t is also determined by sequence of tokens $w_{t+1:n} = [w_{t+1}, ..., w_{t+n}]$. We follow the masking procedure described in the original paper by Devlin *et al.* (2018), where 15 percent of words are randomly designated as targets for prediction, out of which 80 percent are replaced by a masked token ($< mask >$), 10 percent are replaced by a random word and 10 percent remain intact.

The final output of the model is a softmax probability distribution calculated over the entire vocabulary, containing the predicted probabilities of appearance (P) for each word given its left (and in case of the *masked language modelling objective* also right) context. Training therefore consists of the minimization of the negative log-loss (NLL) on the batches of training corpus word sequences by backpropagation through time:

$$\mathrm{NLL} = -\sum_{i=1}^{n} \log P(w_i | w_{1:i-1}) \tag{1}$$

While the *masked language modelling* objective might outperform autoregressive language modelling objective in a setting where a large pretraining corpus is available (Devlin *et al.* 2018) due to the inclusion of the right context, these two training objectives have at least to our knowledge never been compared in a setting where only a relatively small domain specific corpus is available for the pretraining phase. For more details about the performance comparison of these two pretraining objectives, see Section 6.

3.3 Keyword identification

Since each word in the sequence can either be a keyword (or at least part of the keyphrase) or not, the keyword tagging task can be modeled as a binary classification task, where the model is trained to predict if a word in the sequence is a keyword or not.[c] Figure 2 shows an example of how an input text is first transformed into a numerical sequence that is used as an input of the model, which is then trained to produce a sequence of zeroes and ones, where the positions of ones indicate the positions of keywords in the input text.

Since a large majority of words in the sequence are not keywords, the usage of a standard NLL function (see equation 1), which would simply calculate a sum of log probabilities that a word is either a keyword or not for every input word sequence, would badly affect the recall of the model since the majority negative class would prevail. To solve this problem and maximize the recall of the system, we propose a custom classification loss function, where probabilities for each word in the sequence are first aggregated into two distinct sets, one for each class. For example, text "*The advantage of this is to include distributed interactions between the UDDI clients.*" in Figure 2 would be split into two sets, first one containing probabilities for all the words in the input example which are not keywords (*The, advantage, of, this, is, to, include, between, the, clients, .*), and the other containing probabilities for all the words in the input example that are keywords or part of keyphrases (*distributed, interactions, UDDI*). Two NLLs are calculated, one for each probability set, and both are normalized with the size of the set. Finally, the NLLs are summed. More formally, the loss is computed as follows. Let $W = \{w_i\}_{i=1}^{n}$ represent an enumerated sequence of tokens for which predictions are obtained. Let p_i represent the predicted probabilities for the i-th token that it either belongs or does not belong to the ground truth class. The o_i represents the output weight vector of the neural network for token i and j corresponds to the number of classes (two in our

[c] Note that this differs from the sequence labelling approach proposed by Sahrawat *et al.* (2020), where each word in the document is assigned one of three possible labels (see Section 2 for details).

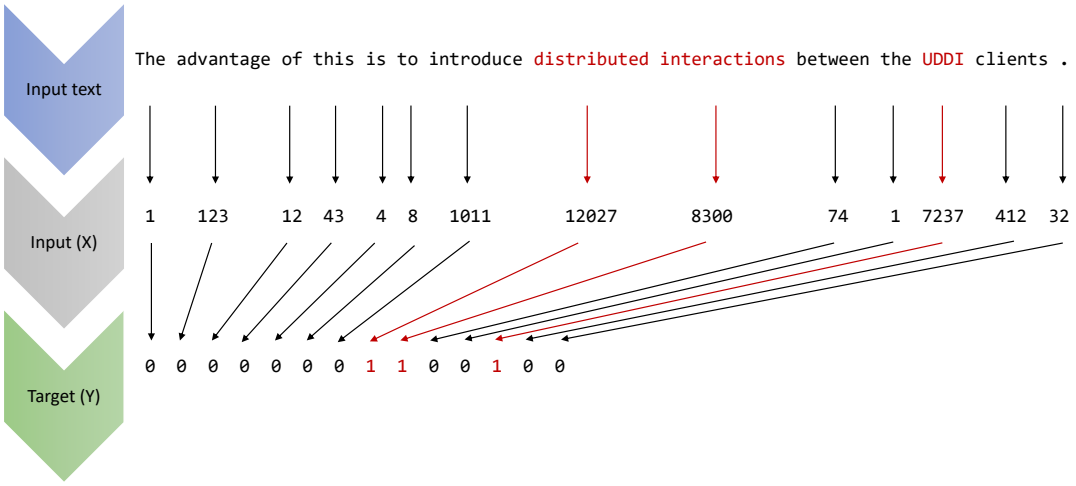


Figure 2: Encoding of the input text "*The advantage of this is to introduce distributed interactions between the UDDI clients.*" with keywords *distributed interactions* and *UDDI*. In the first step, the text is converted into a numerical sequence, which is used as an input to the model. The model is trained to convert this numerical sequence into a sequence of zeroes and ones, where the ones indicate the position of a keyword.

case as the word can be a keyword or not). Predictions are in this work obtained via a log-softmax transform (*lso*), defined as follows (for the i-th token):

$$p_i = \mathrm{lso}(o_i) = \log \frac{\exp(o_i)}{\sum_j \exp(o_j)}.$$

The loss function is comprised from two main parts. Let $K_+ \subseteq W$ represent tokens that are keywords and $K_- \subseteq W$ the set of tokens that are **not** keywords. Note that $|K_- \cup K_+| = n$, i.e., the two sets cover all considered tokens for which predictions are obtained. During loss computation, only the probabilities of the ground truth class are considered. We mark them with p_i^+ or p_i^-. Then the loss is computed as

$$L_+ = -\frac{1}{|K_+|} \sum_{w_i \in K_+} p_i^+ \quad \text{and} \quad L_- = -\frac{1}{|K_-|} \sum_{w_i \in K_-} p_i^-.$$

The final loss is finally computed as:

$$\mathrm{Loss} = L_+ + L_-.$$

Note that even though all predictions are given as an argument, the two parts of the loss address different token indices (i).

In order to produce final set of keywords for each document, tagged words are extracted from the text and duplicates are removed. Note that a sequence of ones is always interpreted as a multi-word keyphrase and not as a combination of one-worded keywords (e.g., *distributed interactions*

from Figure 2 is considered as a single multi-word keyphrase and not as two distinct one word keywords). After that, the following filtering is conducted:

- If a keyphrase is longer than four words, it is discarded.
- Keywords containing punctuation (with the exception of dashes and apostrophes) are removed.
- The detected keyphrases are ranked and arranged according to the softmax probability assigned by the model in a descending order.

3.4 Evaluation

To asses the performance of the model, we measure F1@k score, a harmonic mean between Precision@k and Recall@k.

In a ranking task, we are interested in precision at rank k. This means that only the keywords ranked equal to or better than k are considered and the rest are disregarded. Precision is the ratio of the number of correct keywords returned by the system divided by the number of all keywords returned by the system, or more formally:

$$precision = \frac{|correct\ \ returned\ \ keywords@k|}{|returned\ \ keywords|}$$

Recall@k is the ratio of the number of correct keywords returned by the system and ranked equal to or better than k divided by the number of correct ground truth keywords:

$$recall = \frac{|correct\ \ returned\ \ keywords@k|}{|correct\ \ keywords|}$$

Due to the high variance of a number of ground truth keywords, this type of recall becomes problematic if k is smaller than the number of ground truth keywords, since it becomes impossible for the system to achieve a perfect recall. (Similar can happen to precision@k, if the number of keywords in a gold standard is lower than k, and returned number of keywords is fixed at k.) We shall discuss how this affects different keyword detection systems in Section 7.

Finally, we formally define F1@k as a harmonic mean between Precision@k and Recall@k:

$$F1@k = 2 * \frac{P@k * R@k}{P@k + R@k}$$

In order to compare the results of our approach to other state-of-the-art approaches, we use the same evaluation methodology as Yuan *et al.* (2019) and Meng *et al.* (2019), and measure F1@k with k being either 5 or 10. Note that F1@k is calculated as a harmonic mean of macro-averaged precision and recall, meaning that precision and recall scores for each document are averaged and the F1 score is calculated from these averages. Same as in the related work, lowercasing and stemming are performed on both the gold standard and the generated keywords (keyphrases) during the evaluation. Only keywords that appear in the text of the documents (present keywords)[d] were used as a gold standard and the documents containing no present keywords were removed, in order to make the results of the conducted experiments comparable with the reported results from the related work.

[d]Note that scientific and news articles often list keywords that do not appear in the text of the article. For example, an NLP paper would often list "*Text mining*" as a keyword of the paper, even though the actual phrase does not appear in the text of the paper.

4. Experiments

We first present the datasets used in the experiments. This is followed by the experimental design and the results achieved by TNT-KID in comparison to the state-of-the-art.

4.1 Keyword extraction datasets

Experiments were conducted on seven datasets from two distinct genres, scientific papers about computer science and news. The following datasets from the computer science domain are used:

- **KP20k (Meng *et al.* 2017)**: This dataset contains titles, abstracts, and keyphrases of 570,000 scientific articles from the field of computer science. The dataset is split into train set (530,000), validation set (20,000) and test set (20,000).
- **Inspec (Hulth 2003)**: The dataset contains 2,000 abstracts of scientific journal papers in computer science collected between 1998 and 2002. Two sets of keywords are assigned to each document, the controlled keywords that appear in the Inspec thesaurus, and the uncontrolled keywords, which are assigned by the editors. Only uncontrolled keywords are used in the evaluation, same as by Meng *et al.* (2017), and the dataset is split into 500 test papers and 1500 train papers.
- **Krapivin (Krapivin *et al.* 2009)**: This dataset contains 2,304 full scientific papers from computer science domain published by ACM between 2003 and 2005 with author-assigned keyphrases. 460 papers from the dataset are used as a test set and the others are used for training. Only titles and abstracts are used in our experiments.
- **NUS (Nguyen & Kan 2007)**: The dataset contains titles and abstracts of 211 scientific conference papers from the computer science domain and contains a set of keywords assigned by student volunters and a set of author assigned keywords, which are both used in evaluation.
- **SemEval (Kim *et al.* 2010)**: The dataset used in the SemEval-2010 Task 5, Automatic Keyphrase Extraction from Scientific Articles, contains 244 articles from the computer science domain collected from the ACM Digital Library. 100 articles are used for testing and the rest are used for training. Again, only titles and abstracts are used in our experiments, the article's content was discarded.

From the news domain, three datasets with manually labeled gold standard keywords are used:

- **KPTimes (Gallina *et al.* 2019)**: The corpus contains 279,923 news articles containing editor assigned keywords that were collected by crawling New York Times news website[e]. After that, the dataset was randomly divided into training (92.8 percent), development (3.6 percent) and test (3.6 percent) sets.
- **JPTimes (Gallina *et al.* 2019)**: Similar as **KPTimes**, the corpus was collected by crawling Japan Times online news portal[f]. The corpus only contains 10,000 English news articles and is used in our experiments as a test set for the classifiers trained on the **KPTimes** dataset.
- **DUC (Wan & Xiao 2008)**: The dataset consists of 308 English news articles and contains 2,488 hand labeled keyphrases.

The statistics about the datasets that are used for training and testing of our models are presented in Table 1. Note that there is a big variation in dataset sizes in terms of number of documents (column *No. docs*), and in an average number of keywords (column *Avg. kw.*) and present keywords per document (columns *Avg. present kw.*), ranging from 2.35 present keywords per document in *KPTimes-valid* to 7.79 in *DUC-test*.

[e] `https://www.nytimes.com`
[f] `https://www.japantimes.co.jp`

Table 1. : Datasets used for empirical evaluation of keyword extraction algorithms. *No.docs* stands for number of documents, *Avg. doc. length* stands for average document length in the corpus, *Avg. kw.* stands for average number of keywords per document in the corpus, *% present kw.* stands for the percentage of keywords that appear in the corpus (i.e., percentage of document's keywords that appear in the text of the document) and *Avg. present kw.* stands for the average number of keywords per document that actually appear in the text of the specific document.

Dataset	No. docs	Avg. doc. length	Avg. kw.	% present kw.	Avg. present kw.
Computer science papers					
KP20k-train	530,000	156.34	5.27	62.43	3.29
KP20k-valid	20,000	156.55	5.26	62.30	3.28
KP20k-test	20,000	156.52	5.26	62.55	3.29
Inspec-valid	1500	125.21	9.57	76.92	7.36
Inspec-test	500	121.82	9.83	78.14	7.68
Krapivin-valid	1844	156.65	5.24	54.34	2.85
Krapivin-test	460	157.76	5.74	55.66	3.20
NUS-test	211	164.80	11.66	50.47	5.89
SemEval-valid	144	166.86	15.67	45.43	7.12
SemEval-test	100	183.71	15.07	44.53	6.71
News articles					
KPTimes-train	259,923	783.32	5.03	47.30	2.38
KPTimes-valid	10,000	784.65	5.02	46.78	2.35
KPTimes-test	10,000	783.47	5.04	47.59	2.40
JPTimes-test	10,000	503.00	5.03	76.73	3.86
DUC-test	308	683.14	8.06	96.62	7.79

4.2 Experimental design

We conducted experiments on the datasets described in Section 4.1. First, we lowercased and tokenized all datasets. We experimented with two tokenization schemes, word tokenization and Sentencepiece (Kudo & Richardson 2018) byte-pair encoding (see Section 6 for more details on how these two tokenization schemes affect the overall performance). During both tokenization schemes, a special $< eos >$ token is used to indicate the end of each sentence. For the best performing model, for which the results are presented in Section 4.3, byte-pair encoding was used. For generating the additional POS tag sequence input described in Section 3.1, which was **not** used in the best performing model, Averaged Perceptron Tagger from the NLTK library (Loper & Bird 2002) was used. The neural architecture was implemented in PyTorch (Paszke *et al.* 2019).

In the pretraining phase, two language models were trained for up to ten epochs, one on the concatenation of all the texts from the computer science domain and the other on the concatenation of all the texts from the news domain. Overall the language model train set for computer science domain contained around 87 million tokens and the news train set about 232 million tokens. These small sizes of the language model train sets enable relatively fast training and smaller model sizes (in terms of number of parameters) due to the reduced vocabulary.

After the pretraining phase, the trained language models were fine-tuned on each dataset's *validation* sets (see Table 1), which were randomly split into 80 percent of documents used for training and 20 percent of documents used for validation. The documents containing more than 256 tokens are truncated, while the documents containing less than 256 tokens are padded with a special $< pad >$ token at the end. Each model was fine-tuned for a maximum of 10 epochs and after each epoch the trained model was tested on the documents chosen for validation. The model

that showed the best performance on this set of validation documents (in terms of F@10 score) was used for keyword detection on the test set. Validation sets were also used to determine the best hyperparameters of the model and all combinations of the following hyperparameter values were tested before choosing the best combination, which is written in bold in the list below and on average worked best for all the datasets in both domains[g]:

- Learning rates: 0.00005, 0.0001, **0.0003**, 0.0005, 0.001
- Embedding size: 256, **512**
- Number of attention heads: 4, **8**, 12
- Sequence length: 128, **256**
- Number of attention layers: 4, **8**, 12

Note that in our experiments, we use the same splits as in related work (Meng *et al.* 2019 2017; Gallina *et al.* 2019) for all datasets with predefined splits (see Table 1). The exceptions are NUS, DUC and JPTimes datasets with no available predefined validation-test splits. For NUS and DUC, 10-fold cross-validation is used and the model used for keyword detection on the JPTimes-test dataset was fine-tuned on the KPTimes-valid dataset. Another thing to consider is that in the related work by Yuan *et al.* (2019), Meng *et al.* (2017) and Gallina *et al.* (2019), to which we are comparing, large datasets KPTimes-train and KP20k-train with 530,000 documents and 260,00 documents, respectively, are used for the classification model training and these trained models are applied on all test sets from the matching domain. On the other hand, we do not train our classification models on these two large train sets but instead use smaller KPTimes-valid and KP20k-valid datasets for training, since we argue that, due to language model pretraining, fine-tuning the model on a relatively small labeled dataset is sufficient for the model to achieve competitive performance. We do however conduct the language model pretraining on the concatenation of all the texts from the computer science domain and the news domain as explained above, and these two corpora also contain texts from KPTimes-train and KP20k-train datasets.

4.3 Keyword extraction results and comparison to the state-of-the-art

In Table 2, we present the results achieved by TNT-KID and a number of algorithms from the related work on the datasets presented in Table 1. Evaluation measures were presented in Section 3.4. Only keywords which appear in a text (present keywords) were used as a gold standard in order to make the results of the conducted experiments comparable with reported results from the related work. Note that TfIdf, TextRank, YAKE and RaKUn algorithms are unsupervised and do not require any training, KEA, Maui, GPT-2, GPT-2 + BiLSTM-CRF and TNT-KID were trained on the different *validation* set for each of the datasets, and CopyRNN and CatSeqD were trained on the large KP20k-train dataset for keyword detection on computer science domain, and on the KPTimes-train dataset for keyword detection on the news domain, since they require a large train set for competitive performance.

For RaKUn (Škrlj *et al.* 2019) and YAKE (Campos *et al.* 2020) we report results for default hyperparameter settings, since the authors of RaKUn, as well as YAKE's authors claim that a single hyperparameter set can offer sufficient performance across multiple datasets. We used the author's official github implementations[h] in the experiments. For KEA and Maui we do not conduct additional testing on corpora for which results are not available in the related work (KPTimes, JPTimes and DUC corpus) due to bad performance of the algorithms on all the corpora for which results are available. Finally, for TfIdf and TextRank we report results from the related work where available (Yuan *et al.* 2019) and use the implementation of the algorithms from the Python Keyphrase

[g]Note that the same set of hyperparameters are also used in the pretraining phase.

[h]`https://github.com/SkBlaz/rakun` and `https://github.com/LIAAD/yake`

Extraction (PKE) library[i] to generate unavailable results. Same as for RaKUn and YAKE, default hyperparameters are used.

For KEA, Maui, CopyRNN and CatSeqD, we report results for the computer science domain published in Yuan *et al.* (2019) and for the news domain we report results for CopyRNN published in Gallina *et al.* (2019). The results that were not reported in the related work are results for CatSeqD on KPTimes, JPTimes and DUC, since this model was originally not tested on these three datasets, and the F1@5 score results for CopyRNN on KPTimes and JPTimes. Again, author's official github implementations[j] were used for training and testing of both models. The models were trained and tested on the large KPTimes-train dataset with a help of a script supplied by the authors of the papers. Same hyperparameters that were used for KP20k training in the original papers (Yuan *et al.* 2019; Meng *et al.* 2019) were used.

We also report results for the unmodified pretrained GPT-2 (Radford *et al.* 2019) model with a standard feed forward token classification head, and a pretrained GPT-2 with a BiLSTM-CRF token classification head, as proposed in Sahrawat *et al.* (2020) and described in Section 3.1[k]. For these two models, we apply the same fine-tuning regime as for TNT-KID, i.e. we fine-tune the models for up to 10 epoch on each dataset's *validation* sets (see Table 1), which were randomly split into 80 percent of documents used for training and 20 percent of documents used for validation. The model that showed the best performance on this set of validation documents (in terms of F@10 score) was used for keyword detection on the test set. We use the default hyperparameters for both models and the original GPT-2 tokenization regime.

Overall, supervised neural network approaches drastically outperform all other approaches. Among them, TNT-KID performs the best on all eight datasets in terms of F1@10 but is outperformed by CatSeqD (on four datasets) or GPT-2+ BiLSTM-CRF (on two datasets) on six out of eight datasets in terms of F1@5. In terms of F1@10, CatSeqD performs competitively on KP20k, Krapivin, NUS, SemEval and KPTimes datasets but is outperformed by a large margin on three other datasets by both GPT-2 + BiLSTM-CRF and TNT-KID. To be more specific, in terms of F1@10, TNT-KID outperforms the CatSeqD approach by almost 20 percentage points on the Inspec dataset, on the DUC dataset, it outperforms CatSeqD by about 25 percentage points, and on JPTimes it outperforms CatSeqD by about 12 percentage points.

While the results of CopyRNN are in a large majority of cases very consistent with CatSeqD (CopyRNN performs slightly better than CatSeqD on DUC and JPTimes, and slightly worse on the other six datasets), results of TNT-KID are very similar to the results of GPT-2 + BiLSTM-CRF. In the majority of cases TNT-KID outperforms GPT-2 + BiLSTM-CRF by a small margin according to both criteria, the exceptions being Inspec and JPTimes, where GPT-2 + BiLSTM-CRF performs the best out of all approaches according to F1@5. Another exception is the SemEval dataset, where the GPT-2 + BiLSTM-CRF is outperformed by TNT-KID by a large margin of about 12 percentage points. On the other hand, a GPT-2 model with a standard token classification head does not perform competitively on most datasets.

When it comes to the F1@5 measure, TNT-KID performs competitively on all the datasets. It outperforms all other algorithms on two datasets (KPTimes and DUC) and on average still performs the best out of all algorithms (see row *average*). Nevertheless, the performance in terms of F1@5 is still noticeably worse than in terms of F1@10. The difference between TNT-KID and CatSeqD, which performs the best on four out of eight datasets in terms of F1@5, can be partially explained by the difference in training regimes and the fact that our system was designed to maximize recall (see Section 3). Since our system generally detects more keywords than CatSeqD and CopyRNN, it tends to achieve better recall, which offers a better performance when up to

[i] `https://github.com/boudinfl/pke`

[j] `https://github.com/memray/OpenNMT-kpg-release`

[k] We use the implementation of GPT-2 from the Transformers library (`https://github.com/huggingface/transformers`) and use the Pytorch-crf library (`https://pytorch-crf.readthedocs.io/en/stable/`) for the implementation of the BiLSTM-CRF token classification head.

Table 2. : Empirical evaluation of state-of-the-art keyword extractors. Results marked with * were obtained by our implementation or reimplementation of the algorithm and results without * were reported in the related work.

	Unsupervised approaches				Supervised approaches						
	TfIdf	TextRank	YAKE	RaKUn	KEA	Maui	CopyRNN	CatSeqD	GPT-2	GPT-2 + BiLSTM-CRF	TNT-KID
					KP20k						
F1@5	0.072	0.181	0.141*	0.177*	0.046	0.005	0.317	**0.348**	0.252*	0.339*	0.342*
F1@10	0.094	0.151	0.146*	0.160*	0.044	0.005	0.273	0.298	0.256*	0.342*	**0.346***
					Inspec						
F1@5	0.160	0.286	0.204*	0.101*	0.022	0.035	0.244	0.276	0.293*	**0.467***	0.447*
F1@10	0.244	0.339	0.223*	0.108*	0.022	0.046	0.289	0.333	0.335*	**0.525***	**0.525***
					Krapivin						
F1@5	0.067	0.185	0.215*	0.127*	0.018	0.005	0.305	**0.325**	0.210*	0.280*	0.301*
F1@10	0.093	0.160	0.196*	0.106*	0.017	0.007	0.266	0.285	0.214*	0.283*	**0.307***
					NUS						
F1@5	0.112	0.230	0.159*	0.224*	0.073	0.004	0.376	**0.374**	0.274*	0.311*	0.350*
F1@10	0.140	0.216	0.196*	0.193*	0.071	0.006	0.352	0.366	0.305*	0.332*	**0.369***
					SemEval						
F1@5	0.088	0.217	0.151*	0.167*	0.068	0.011	0.318	**0.327**	0.261*	0.214	0.291*
F1@10	0.147	0.226	0.212*	0.159*	0.065	0.014	0.318	0.352	0.295*	0.232	**0.355***
					KPTimes						
F1@5	0.179*	0.022*	0.105*	0.168*	*	*	0.406*	0.424*	0.353*	0.439*	**0.469***
F1@10	0.151*	0.030*	0.118*	0.139*	*	*	0.393	0.424*	0.354*	0.440*	**0.469***
					JPTimes						
F1@5	0.266*	0.012*	0.109*	0.225*	*	*	0.256*	0.238*	0.258*	**0.344***	0.337*
F1@10	0.229*	0.026*	0.135*	0.185*	*	*	0.246	0.238*	0.267*	0.346*	**0.360***
					DUC						
F1@5	0.098*	0.120*	0.106*	0.189*	*	*	0.083	0.063*	0.247*	0.281*	**0.312***
F1@10	0.120*	0.181*	0.132*	0.172*	*	*	0.105	0.063*	0.277*	0.321*	**0.355***
					Average						
F1@5	0.130	0.157	0.149	0.172	*	*	0.288	0.297	0.269*	0.334*	**0.356***
F1@10	0.152	0.166	0.170	0.153	*	*	0.280	0.295	0.288*	0.353*	**0.386***

10 keywords need to be predicted. On the other hand, a more conservative system that generally predicts less keywords tends to achieve a better precision, which positively affects the F1 score in a setting where only up to 5 keywords need to be predicted. This phenomenon will be analysed in more detail in Section 5, where we also discuss the very low results achieved by CopyRNN and CatSeqD on the DUC dataset.

When it comes to two other supervised approaches, KEA and Maui, they perform badly on all datasets they have been tested on and are outperformed by a large margin even by all unsupervised approaches. When we compare just unsupervised approaches, TextRank achieves by far the best results according to both measures on the Inspec dataset. This is the dataset with the on average shortest documents. On the other hand, TextRank performs uncompetitively in comparison to other unsupervised approaches on two datasets with much longer documents, KPTimes and JPTimes, where RaKUn and TfIdf are the best unsupervised approaches, respectively. Interestingly, it achieves the highest F@10 score out of all unsupervised keyword detectors on the DUC dataset, which also contains long documents. Perhaps this could be explained by the average number of present keywords, which is much higher for DUC-test (7.79) than for KPTimes-test (2.4) and JPTimes-test (3.86) datasets.

Overall (see row *average*), TNT-KID offers the most robust performance on the test datasets and is closely followed by GPT-2 + BiLSTM. CopyRNN and CatSeqD are very close to each

other according to both criteria. Out of unsupervised approaches, on average all of them offer surprisingly similar performance. According to the F@10 score, YAKE on average works slightly better than the second ranked TextRank and also in general offers more steady performance, since it gives the most consistent results on a variety of different datasets. Similar could be said for RaKUn, the best ranked unsupervised algorithm according to the F@5 score.

Examples of the TNT-KID keyword detection are presented in the Appendix.

5. Error analysis

In this Section we first analyse the reasons, why transformer based TNT-KID is capable of outperforming other state-of-the-art neural keyword detectors, which employ a generative model, by a large margin on some of the datasets. Secondly, we gather some insights into the inner workings of the TNT-KID by a visual analysis of the attention mechanism.

5.1 Comparison between TNT-KID and CatSeqD

As was observed in Section 4.3, transformer based TNT-KID and GPT-2 + BiLSTM-CRF outperform generative models CatSeqD and CopyRNN by a large margin on the Inspec, JPTimes and DUC datasets. Here, we try to explain this discrepancy by focusing on the difference in performance between the best transformer based model, TNT-KID, and the best generative model, CatSeqD. The first hypothesis is connected with the statistical properties of the datasets used for training and testing, or more specifically, with the average number of keywords per document for each dataset. Note that CatSeqD is trained on the KP20k-train, when employed on the computer science domain, and on the KPTimes-train dataset, when employed on news. Table 1 shows that both of these datasets do not contain many present keywords per document (KP20k-train 3.28 and KPTimes-train 2.38), therefore training the model on these datasets conditions it to be conservative in its predictions and to assign less keywords to each document than a more liberal TNT-KID. This gives the TNT-KID a competitive advantage on the datasets with more present keywords per document.

Figure 3 shows a correlation between the average number of present keywords per document for each dataset and the difference in performance in terms of F@10, measured as a difference between an F@10 score achieved by TNT-KID and an F@10 score achieved by CatSeqD. The difference in performance is the biggest for the DUC dataset (about 30 percentage points) that on average has the most keywords per document, 7.79, and second biggest for Inspec, in which an average document has 7.68 present keywords.

The above hypothesis explains why CatSeqD offers competitive performance on the KP20k-test, Krapivin-test, NUS-test and KPTimes-test datasets with similar number of keywords per document than its two train sets but does not explain the competitive performance of CatSeqD on the SemEval test set that has 6.71 present keywords per document. Even more importantly, it does not explain the large difference in performance between TNT-KID and CatSeqD on the JPTimes-test. This suggests that there is another factor influencing the performance of some keyword detectors.

The second hypothesis suggests that the difference in performance could be explained by the difference in training regimes and the different tactics used for keyword detection by the two systems. While TNT-KID is fine-tuned on each of the datasets, no fine-tuning is conducted for CatSeqD that needs to rely only on the information obtained during training on the large KP20k-train and KPTimes-train datasets. This information seems sufficient when CatSeqD is tested on datasets that contain similar keywords than the train sets. On the other hand, this training regime does not work for datasets that have less overlapping keywords.

Figure 4 supports this hypothesis by showing strong correlation between the difference in performance in terms of F@10 and the percentage of keywords that appear both in the CatSeqD train sets (KP20k-train and KPTimes-train for computer science and news domain, respectively) and the test datasets. DUC and Inspec datasets have the smallest overlap, with only 17 percent of keywords in DUC appearing in the KPTimes-train and with 48 percent of keywords in Inspec appearing in the KP20k-train set. On the other hand, Krapivin, NUS, KP20k and KPTimes, the test sets on which CatSeqD performs more competitively, are the datasets with the biggest overlap, reaching up to 95 percent for KPTimes-test.

Figure 4 also explains a relatively bad performance of CatSeqD on the JPTimes corpus (see Table 2) despite the smaller average number of keywords per document. Interestingly, despite the fact that no dataset specific fine-tuning for TNT-KID is conducted on the JPTimes corpus (since there is no validation set available, fine-tuning is conducted on the KPTimes-valid), TNT-KID manages to outperform CatSeqD on this dataset by about 12 percentage points. This suggests that a smaller keyword overlap between train and test sets has less of an influence on the TNT-KID and could be explained with the fact, that CatSeqD considers keyword extraction as a generation task and tries to generate a correct keyword sequence, while TNT-KID only needs to tag an already existing word sequence, which is an easier problem that perhaps requires less specific information gained during training.

According to the Figure 4, the SemEval test set is again somewhat of an outlier. Despite the keyword overlap that is quite similar to the one in the JPTimes test set and despite having a relatively large set of present keywords per document, CatSeqD still performs competitively on this corpus. This points to a hypothesis that there might be another unidentified factor, either negatively influencing the performance of TNT-KID and positively influencing the performance of CatSeqD, or the other way around.

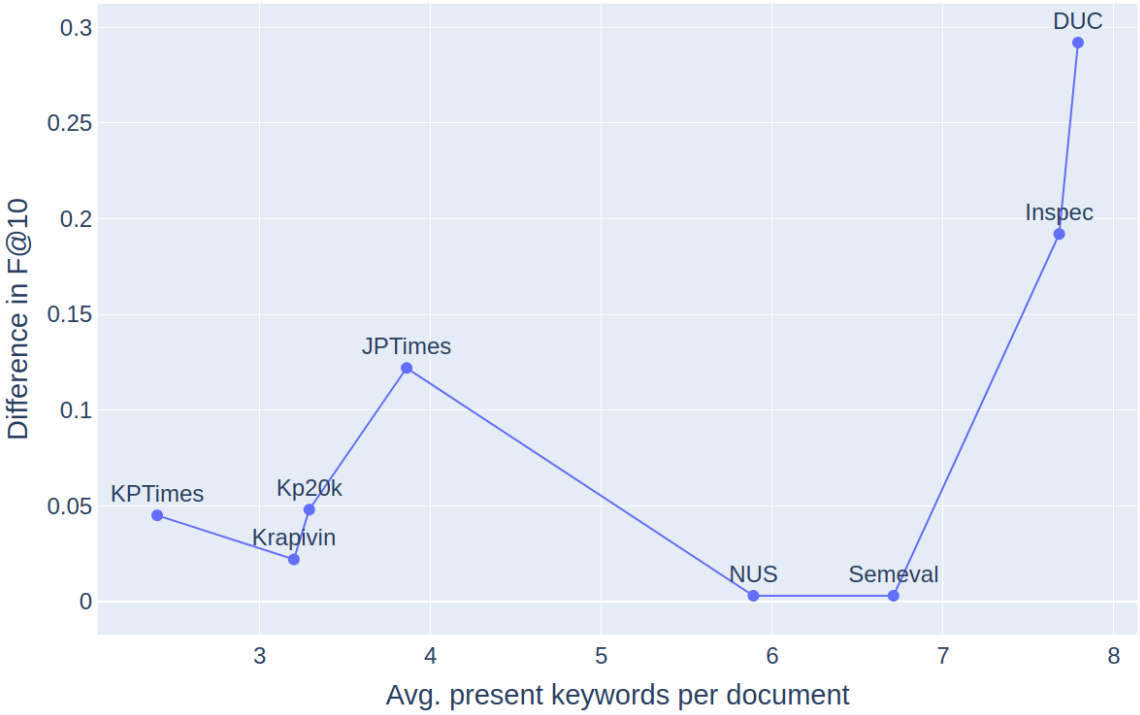


Figure 3: Relation between the average number of present keywords per document for each test dataset and the difference in performance ($F@10_{\text{TNT-KID}} - F@10_{\text{CatSeqD}}$).

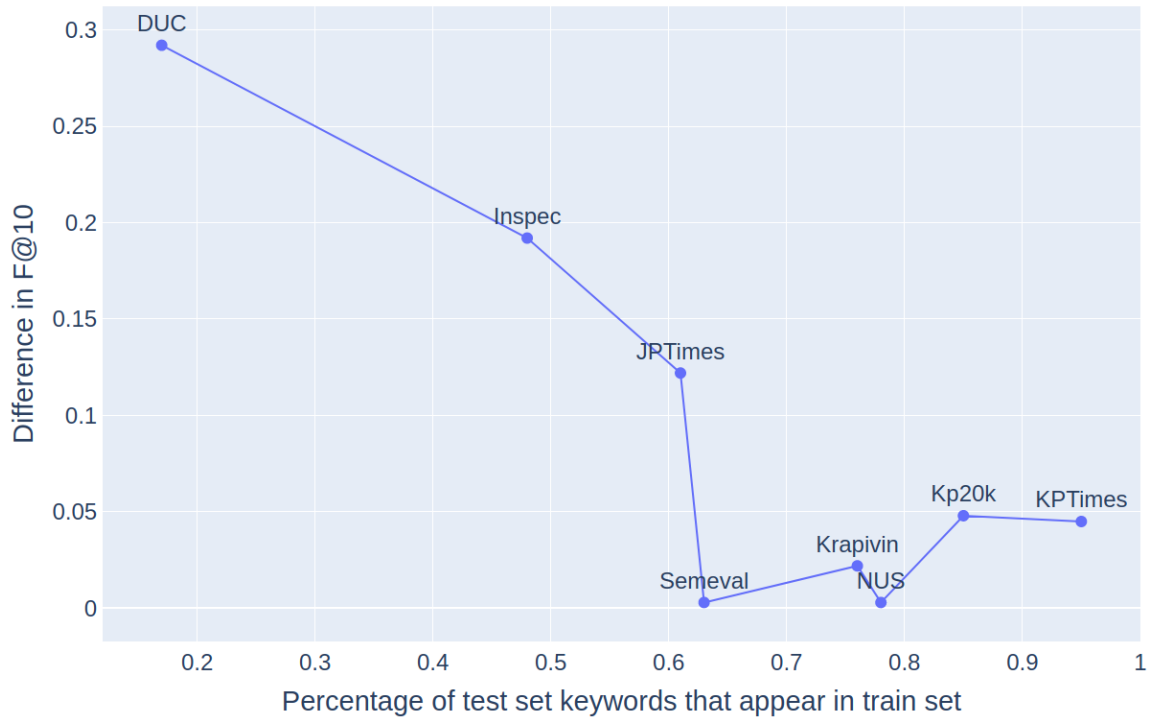


Figure 4: Relation between the percentage of keywords that appear in the train set for each test dataset and the difference in performance ($F@10_{\text{TNT-KID}} - F@10_{\text{CatSeqD}}$).

5.2 CatSeqD fine-tuning

According to the results in Section 4.3, supervised approaches to the keyword extraction task tend to outperform unsupervised approaches, most likely due to their ability to adapt to the specifics of the syntax, semantics and keyword labeling regime of the specific corpus. On the other hand, the main disadvantage of most supervised approaches is that they require a large dataset with labeled keywords for training, which are scarce at least in some languages. In this paper we argue, that the main advantage of the proposed TNT-KID approach is, that due to its language model pretraining, the model only requires a small labeled dataset in order to fine-tune the language model for the keyword classification task. This fine-tuning allows the model to adapt to each dataset and leads to a better performance of TNT-KID in comparison to CatSeqD, for which no fine-tuning was conducted.

Even though no fine-tuning was conducted in the original CatSeqD study (Yuan *et al.* 2019), one might hypothesise that the performance of CatSeqD could be further improved if the model would be fine-tuned on each dataset, same as TNT-KID. To test this hypothesis, we take the CatSeqD model trained on KP20k, conduct additional training on the SemEval, Krapivin and Inspec validation sets (i.e., all datasets besides KP20k and KPTimes with a validation set), and test these fine-tuned models on the corresponding test sets. Fine-tuning was conducted for up to 100.000 train steps[1] and the results are presented in Figure 5.

Only on one of the three datasets, the Inspec test set, the performance can be improved by additional fine-tuning. Though the improvement on the Inspec test set of about 10 percentage points (from 33.5% to 44%) in terms of F1@10 is quite substantial, the model still performs worse than TNT-KID, which achieves F1@10 of 52.5%. The improvement is most likely connected with the fact that the Inspec test set contains more keywords that do not appear in the KP20k than

[1]Same hyperparameters that were used for KP20k training in the original paper (Yuan *et al.* 2019) were used for fine-tuning.

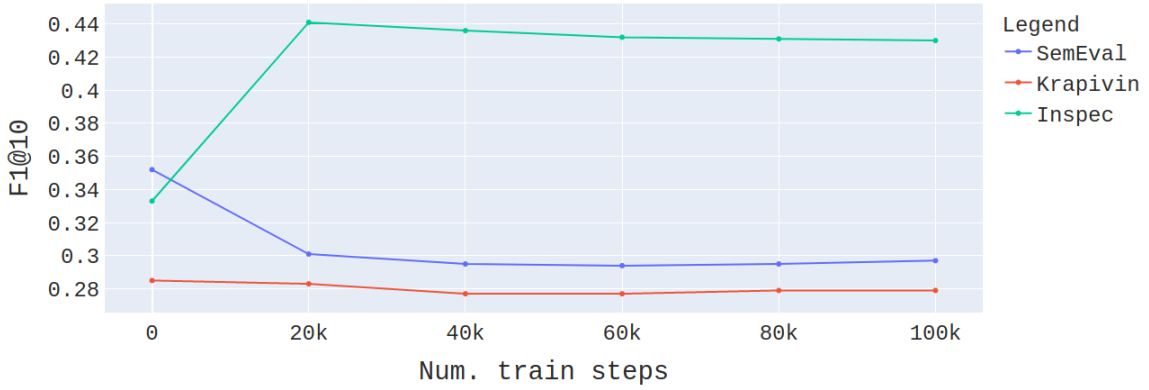


Figure 5: Performance of the KP20k trained CatSeqD model fine-tuned on SemEval, Krapivin and Inspec validation sets and tested on the corresponding test sets, in correlation with the length of the fine tuning in terms of number of train steps. Zero train steps means that the model was not fine-tuned.

SemEval and Krapivin test sets (see Figure 4). Inspec test set also contains more keywords per document than the other two test sets (7.68 present keyword on average, in comparison to 6.71 present keywords per document in the SemEval test set and 3.2 in the Krapivin test set). Since the KP20k train set on average contains only 3.29 present keywords per document, the fine-tuning on the Inspec dataset most likely also adapts the classifier to a more liberal keyword labeling regime.

On the other hand, fine-tuning does not improve the performance on the Krapivin and SemEval datasets. While there is no difference between the fine-tuned and original model on the Krapivin test set, fine-tuning negatively affects the performance of the model on the SemEval dataset. The F1@10 score drops from about 35% to about 30% after 20.000 train steps. Further fine-tuning does not have any effect on the performance. The hypothesis is, that this drop in performance is somewhat correlated with the size of the SemEval validation set, which is much smaller (it contains only 144 documents) than Inspec and Krapivin validation sets (containing 1500 and 1844 documents, respectively), and this causes the model to overfit. Further tests would however need to be conducted to confirm or deny this hypothesis.

Overall, 20.000 train steps seem to be enough for model adaptation in each case, since the results show that additional fine-tuning does not have any influence on the performance.

5.3 Dissecting the attention space

One of the advantages of the transformer architecture is its employment of the attention mechanism, that can be analysed and visualized, offering valuable insights into inner workings of the system and enabling interpretation of how the neural net tackles the keyword identification task. The TNT-KID attention mechanism consists of multiple attention heads (Vaswani *et al.* 2017) – square matrices linking pairs of tokens within a given text – and we explored how this (activated) weight space can be further inspected via visualization and used for interpretation.

While square attention matrices show importance of the correlations between all tokens in the document for a keyword identification task, we focused only on the diagonals of the matrices, which indicate how much attention the model pays to the “correlation” a specific word has with itself, i.e., how important is a specific word for the classification of a specific token as either being a keyword or not. We extracted these diagonal attention scores for eight attention heads of the last out of eight encoders, for each of the documents in the SemEval-test and averaged the scores across an entire dataset by summing together scores belonging to the same position in each head and dividing this sum with the number of documents. Figure 6 shows the average attention score

of each of the eight attention heads for each token position. While there are distinct differences between heads, a distinct peak at the beginning of the attention graph can be observed for all heads but one (head 4), which means that heads generally pay more attention to the tokens at the beginning of the document. This suggests that the system has learned that tokens appearing at the

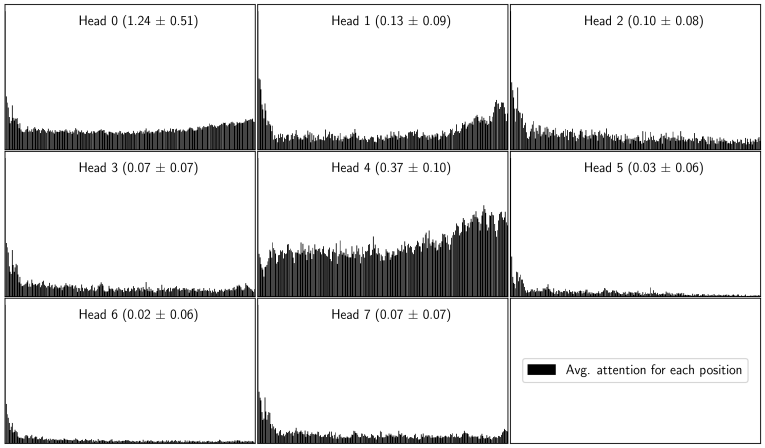


Figure 6: Average attention for each token position in the SemEval corpus across eight attention heads. Distinct peaks can be observed for tokens appearing at the beginning of the document in all but one out of eight attention heads.

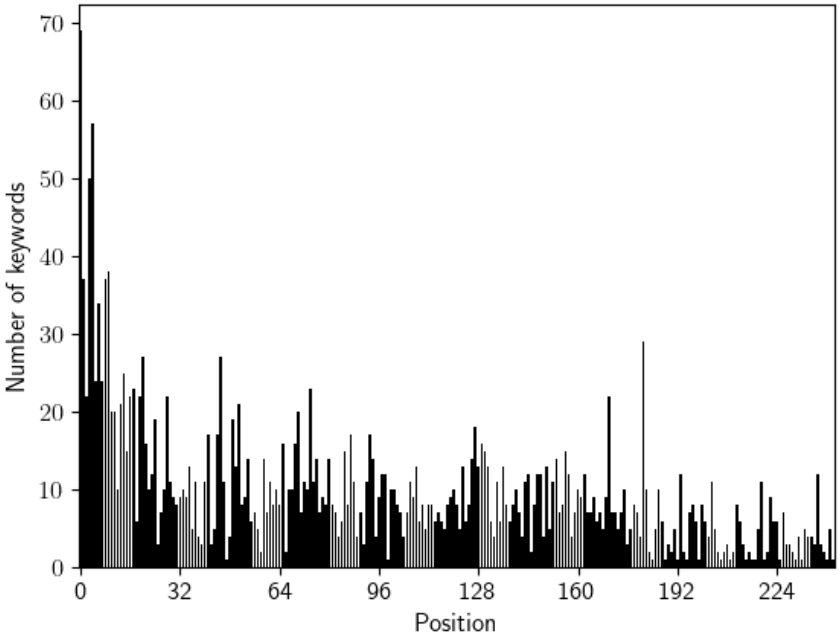


Figure 7: Number of keywords for each token position in the SemEval corpus. Distinct peaks can be observed for positions at the beginning of the document.

beginning of the document are more likely to be keywords (Figure 7 shows the actual keyword count for each position in the SemEval corpus) and once again shows the importance of positional information for the task of keyword identification.

Another insight into how the system works can be gained by analysing how much attention was paid to each individual token in each document. Figure 8 displays attentions for individual tokens, as well as marks them based on predictions for an example document from the SemEval-test. Green tokens were correctly identified as keywords, red tokens were incorrectly identified as keywords and less transparency (more colour) indicates that a specific token received more attention from the classifier.

Figure 8 shows that, at least for this specific document, tokens that were either correctly or incorrectly classified as keywords did receive more attention than an average token. There are also some tokens that received a lot of attention and were not classified as keywords, e.g., *eos* (end of sentence signs) and *pad* (padding) signs, and also words like *of, is, we, etc.*. Another interesting thing to notice is the fact, that the amount of attention associated with individual tokens that appear more than once in the document varies and is somewhat dependent on the position of the token. [m]

[m]Note that Figure 8 is just a motivating example. A more thorough statistical analysis of much more than just one document would be required in order to draw proper conclusions about the behavior of the attention mechanism during keyword identification.

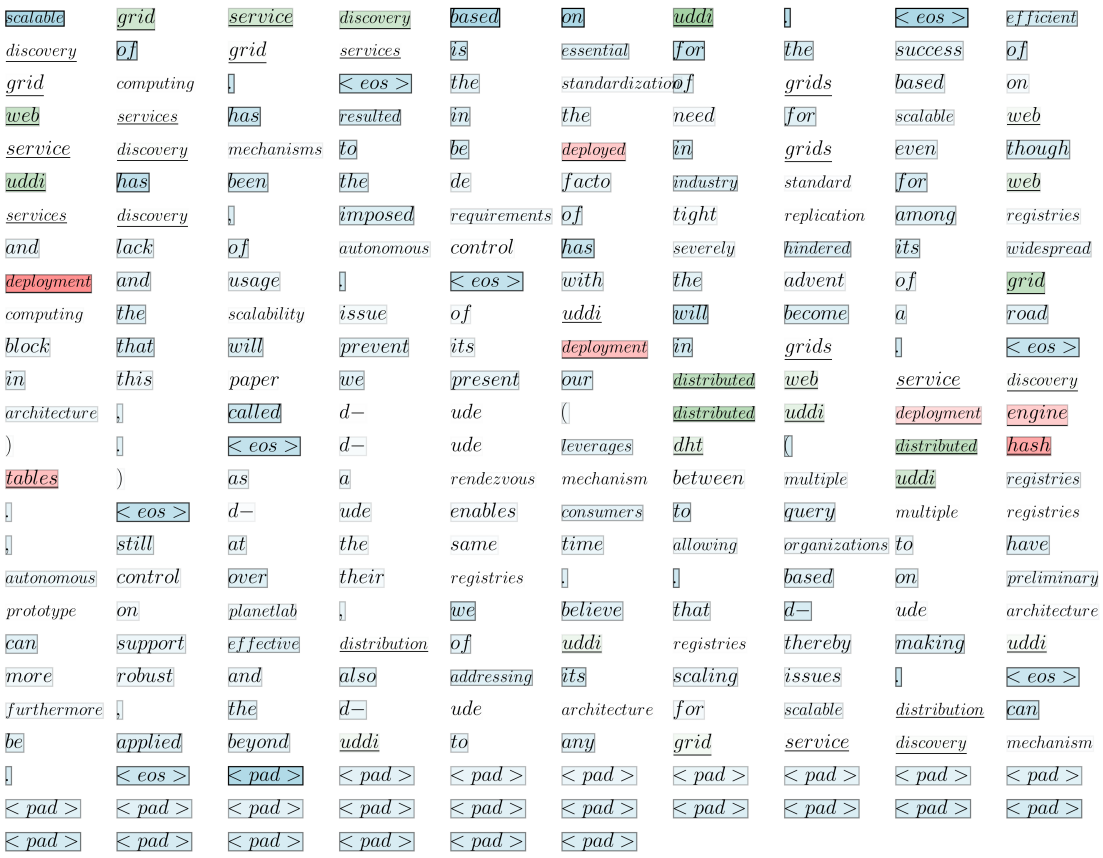


Figure 8: Attention-colored tokens. Green ones were correctly identified as keywords, red ones were incorrectly identified as keywords and less transparency indicates stronger attention for the token. Underlined words in italic were identified as keywords by the system.

6. Ablation study

In this section we explore the influence of several technique choices and building blocks of the keyword extraction workflow on the overall performance of the model:

- **Language model pretraining**; assessment whether pretraining positively affects the performance of the keyword extraction and if the improvements are dataset or domain specific.
- **Choice of pretraining regime**; comparison of two pretraining objectives, autoregressive language modelling and masked language modelling described in Section 3.2.
- **Choice of input tokenization scheme**; comparison of two tokenization schemes, word tokenization and Sentencepiece (Kudo & Richardson 2018) byte-pair encoding.
- **Part-of-speech(POS) tags**; assessment whether adding POS tags as an additional input improves the performance of the model.
- **Transformer architecture adaptations**; as was explained in Section 3.1, in the fine-tuning stage we add an additional BiLSTM encoder to the output of the transformer encoder. We also experiment with the addition of the BiLSTM+CRF token classification head on top of the model, as was proposed in Sahrawat *et al.* (2020) and described in Section 3.1. Here we assess the influence of these additions on the performance of the model.

Table 3 presents results on all datasets for several versions of the model, a model with no language model pretraining (*nolm*), a model pretrained with an autoregressive language model objective (*lm*), a model pretrained with a masked language model objective (*maskedlm*), a model pretrained with an autoregressive language model objective and leveraging byte-pair encoding tokenization scheme (*lm+bpe*), a model pretrained with an autoregressive language model objective and leveraging additional POS tag sequence input (*lm+pos*), a model pretrained with an autoregressive language model objective and a BiLSTM encoder (*lm+rnn*), a model pretrained with an autoregressive language model objective leveraging byte-pair encoding tokenization scheme and a BiLSTM encoder (*lm+bpe+rnn*), and a model pretrained with an autoregressive language model objective leveraging byte-pair encoding tokenization scheme and a BiLSTM+CRF token classification head (*lm+bpe+crf*) .

On average (see last two rows in Table 3), by far the biggest boost in performance is gained by employing the autoregressive language model pretraining (column *lm*), improving the F@5 score by about 10 percentage points and the F@10 score by 11 percentage points in comparison to no language model pretraining (column *nolm*). As expected, the improvements are substantial on three smallest corpora, which by themselves do not contain enough text for the model to obtain sufficient syntactic and semantic knowledge. The largest gains are achieved on the NUS test set, where almost an 84 percent improvement in terms of the F@10 score can be observed, and on the SemEval test set, where the improvement of 78 percent in terms of F@5 can be observed. We also observe about a 47 percent improvement in terms of F@10 on the DUC test set. Not surprisingly, for the KP20k dataset, which has a relatively large validation set used for fine-tuning, we can observe a much smaller improvement of about 23 percent in terms of F@10. On the other hand, we observe a substantial improvement of roughly 50 percent in terms of both F@5 and F@10 on the KPTimes test set, even though the KPTimes validation set used for fine-tuning is the same size as KP20k validation set. This means that in the language modelling phase the model still manages to obtain knowledge that is not reachable in the fine-tuning phase and can perhaps be partially explained by the fact that all documents are truncated into 256 tokens long sequences in the fine-tuning phase. The KPTimes-valid dataset, used both for language modelling and fine-tuning, has on average 784.65 tokens per document, which means that more than half of the document's text is discarded during the fine-tuning phase. This is not the case in the language modelling phase, where all of the text is leveraged.

Table 3. : Results of the ablation study. Column *lm+bpe+rnn* represents the results for the model that was used for comparison with other methods from the related work in Section 4.3.

	nolm	lm	maskedlm	lm+bpe	lm+pos	lm+rnn	**lm+bpe+rnn**	lm+bpe+crf
					KP20k			
F1@5	0.2544	0.2922	0.2476	0.2958	0.3003	0.3349	0.3418	**0.3478**
F1@10	0.2304	0.2836	0.2313	0.2941	0.2986	0.3382	0.3457	**0.3521**
					Inspec			
F1@5	0.2868	0.4099	0.2875	0.4255	0.4136	**0.4506**	0.4471	0.4463
F1@10	0.3636	0.4994	0.3704	0.4871	0.5012	**0.5253**	0.5252	0.5147
					Krapivin			
F1@5	0.1919	0.2277	0.2046	0.2879	0.2494	0.3088	0.3009	**0.3142**
F1@10	0.1904	0.2314	0.2029	0.2884	0.2555	0.3164	0.3070	**0.3178**
					NUS			
F1@5	0.1909	0.3319	0.2372	0.3352	0.3339	0.3419	**0.3502**	0.3371
F1@10	0.1902	0.3492	0.2552	0.3586	0.3518	0.3626	**0.3686**	0.3658
					SemEval			
F1@5	0.1671	0.3070	0.1842	0.2462	0.2780	0.2696	**0.2921**	0.2524
F1@10	0.1950	0.3469	0.2528	0.2913	0.3426	0.3303	**0.3552**	0.3007
					KPTimes			
F1@5	0.2864	0.4242	0.3052	0.4211	0.4306	0.4627	**0.4691**	0.4408
F1@10	0.2760	0.4208	0.3017	0.4208	0.4300	0.4609	**0.4693**	0.4413
					JPTimes			
F1@5	0.2490	0.3305	0.2644	0.3341	0.3359	**0.3790**	0.3570	0.3357
F1@10	0.2478	0.3344	0.2705	0.3373	0.3402	**0.3823**	0.3596	0.3372
					DUC			
F1@5	0.1951	0.2848	0.1523	0.2759	0.2918	0.3003	**0.3115**	0.2943
F1@10	0.2265	0.3340	0.1979	0.3213	0.3386	0.3432	**0.3551**	0.3342
					Average			
F@5	0.2277	0.3260	0.2354	0.3277	0.3292	0.3560	**0.3587**	0.3461
F@10	0.2400	0.3500	0.2603	0.3499	0.3573	0.3824	**0.3857**	0.3705

On the other hand, using the masked language modelling pretraining (column *maskedlm*) objective on average yields only a somewhat negligible improvement of about 0.8 percentage point in terms of F@5 score and a slightly bigger improvement of about 2 percentage points in terms of F@10 score in comparison to no language model pretraining. It does however improve the performance on the two smallest datasets, NUS (by about 6.5 percentage points in terms of F1@10) and SemEval (by about 6 percentage points in terms of F1@10). The large discrepancy in performance between the two different language model objectives can be partially explained by the sizes of the pretraining corpora. By using autoregressive language modelling, the model learns to predict the next word probability distribution for each sequence in the corpus. By using the masked language modelling objective, 15 percent of the words in the corpus are randomly masked and used as targets for which the word probability distributions need to be predicted from the surrounding context. Even though each training epoch a different set of words is randomly masked, it is quite possible, that some words are never masked due to small sizes of the corpora and since we only train the model for up to 10 epochs.

Results show that adding POS tags as an additional input (column *lm+pos*) leads to only marginal performance improvements. Some previous studies suggest that transformer based models that employ transfer learning already capture sufficient amount of syntactic and other information about the composition of the text (Jawahar *et al.* 2019). Our results therefore support the hypothesis that additional POS tag inputs are somewhat unnecessary in the transfer learning setting but additional experiments would be needed to determine whether this is task/language specific or not.

Another adaptation that does not lead to any significant improvements when compared to the column *lm* is the usage of the byte-pair encoding scheme (column *lm+bpe*). The initial hypothesis that motivated the usage of byte-pair encoding was that it might help the model's performance by introducing some knowledge about the word composition and by enabling the model to better understand that different forms of the word can represent the same meaning. However, the usage of byte-pair encoding might on the other hand also negatively affect the performance, since splitting up words inside a specific keyphrase would make these keyphrases longer in terms of number of words and detecting a longer continuous word sequence as a keyword might represent a harder problem for the model than detecting a shorter one. Nevertheless, usage of byte-pair encoding does have an additional positive effect of drastically reducing the vocabulary of the model (e.g., for computer science articles, this means a reduction from about 250.000 tokens to about 30.000) and with it also the number of parameters in the model (from about 290 million to about 70 million).

Adding an additional BiLSTM encoder in the fine-tuning stage of a pretrained model (column *lm+rnn*) leads to consistent improvements on almost all datasets and to an average improvement of about 3 percentage points in terms of both F@5 and F@10 scores. This confirms the findings from the related work that recurrent neural networks work well for the keyword detection task and also explains why a majority of state-of-the-art keyword detection systems leverage recurrent layers.

We also present results for a model in which we employed autoregressive language model pretraining, used byte-pair encoding scheme and added a BiLSTM encoder (column *lm+bpe+rnn*) that was used for comparison with other methods from the related work in Section 4.3, and results for the approach proposed by Sahrawat *et al.* (2020), where a BiLSTM+CRF token classification head is added on top of the transformer encoder, that employs byte-pair encoding scheme and autoregressive language model pretraining (column *lm+bpe+crf*). The BiLSTM+CRF performs quite well, outperforming all other configurations on two (KP20k and Krapivin) datasets. On average it however still performs by more than 1 percentage point worse than both configurations employing an added BiLSTM encoder.

7. Conclusion and future work

In this research we have presented TNT-KID, a novel transformer based neural tagger for keyword identification that leverages a transfer learning approach to enable robust keyword identification on a number of datasets. The presented results show that the proposed model offers a robust performance across a variety of datasets with manually labeled keywords from two different domains. By exploring the differences in performance between our model and the best performing generative model from the related work, CatSeqD by Yuan *et al.* (2019), we manage to pinpoint strengths and weaknesses of each model and therefore enable a potential user to choose the approach most suitable for the task at hand. By visualizing the attention mechanism of the model, we try to interpret classification decisions of the neural network and show that efficient modelling of positional information is essential in the keyword detection task. Finally, we propose an ablation study which shows how specific components of the keyword extraction workflow influence the overall performance of the model.

The biggest advantage of supervised approaches to keyword extraction task is their ability to adapt to the specifics of the syntax, semantics, content, genre and keyword tagging regime of the specific corpus. Our results show that this offers a significant performance boost and state-of-the-art supervised approaches outperform state-of-the-art unsupervised approaches on the majority of datasets. On the other hand, the ability of the supervised models to adapt might become limited in cases when the train dataset is not sufficiently similar to the dataset on which keyword detection needs to be performed. This can clearly be seen on the DUC dataset, in which only about 17 percent of keywords also appear in the KPTimes train set, used for training the generative CopyRNN and CatSeqD models. Here, these two state-of-the-art models perform the worst of all the models tested and as is shown in Section 5.2, this *keywordinees* generalization problem can not be overcome by simply fine-tuning these state-of-the-art systems on each specific dataset.

On the other hand, TNT-KID bypasses the generalization problem by allowing fine-tuning on very small datasets. Nevertheless, the results on the JPTimes corpus suggest that it also generalizes better than CopyRNN and CatSeqD. Even though all three algorithms are trained on the KPTimes dataset (since JPTimes corpus does not have a validation set)[n], TNT-KID manages to outperform the other two by about 10 percentage points according to the F1@10 and F1@5 criteria despite the discrepancy between train and test set keywords. As already mentioned in Section 5.1, this can be partially explained by the difference in approaches used by the models and the fact that keyword generation is a much harder task than keyword tagging. For keyword generation task to be successful, seeing a sequence that needs to be generated in advance, during training, is perhaps more important, than for a much simpler task of keyword tagging, where a model only needs to decide if a word is a keyword or not. Even though the keyword generators try to ease the task by employing a copying mechanism (Gu *et al.* 2016), the experiments suggest that generalizing *keywordinees* to unseen word sequences still represent a bigger challenge for these models than for TNT-KID.

While the conducted experiments suggest that TNT-KID works better than other neural networks in a setting where previously unseen keywords (i.e., keywords not present in the training set) need to be detected, further experiments need to be devised to evaluate the competitiveness of TNT-KID in a cross-domain setting when compared to unsupervised approaches. Therefore, in order to determine if the model's internal representation of *keywordiness* is general enough to be transferable across different domains, in the future we also plan to conduct some cross-domain experiments.

Another aspect worth mentioning is the evaluation regime and how it affects the comparison between the models. By fine-tuning the model on each dataset, the TNT-KID model learns the optimal number of keywords to predict for each specific dataset. This number is in general slightly above the average number of present keywords in the dataset, since the loss function was adapted to maximize recall (see Section 3). On the other hand, CatSeqD and CopyRNN are only trained on the KP20k-train and KPTimes-train datasets that have less present keywords than a majority of test datasets. This means our system on average predicts more keywords per document than these two systems, which negatively affects the precision of the proposed system in comparison to CatSeqD and CopyRNN, especially at smaller k values. On the other hand, predicting less keywords hurts recall, especially on datasets where documents have on average more keywords. As already mentioned in Section 6, this explains why our model compares better to other systems in terms of F@10 than in terms of F@5 and also raises a question how biased these measures of performance actually are. Therefore, in the future we plan to use other performance measures to compare our model to others.

Overall, the differences in training and prediction regimes between TNT-KID and other neural models imply that the choice of a network is somewhat dependent on the use-case. If a large training dataset of an appropriate genre with manually labeled keywords is available and if the

[n] Note that TNT-KID is trained on the validation set, while CopyRNN and CatSeqD are trained on the much larger train set.

system does not need to predict many keywords, then CatSeqD is most likely the best choice, even though TNT-KID shows competitive performance on a large majority of datasets. On the other hand, if only a relatively small train set is available and it is preferable to predict a larger number of keywords, then the results of this study suggest that TNT-KID is most likely a better choice.

The conducted study also indicates that the adaptation of the transformer architecture and the training regime for the task at hand can lead to improvements in keyword detection. Both TNT-KID and a pretrained GPT-2 model with a BiLSTM + CRF token classification head manage to outperform the unmodified GPT-2 with a default token classification head by a large margin. Even more, TNT-KID manages to outperform both, the pretrained GPT-2 and the GPT-2 with BiLSTM + CRF, even though it employs only 8 attention layers, 8 attention heads and an embedding size of 512 instead of the standard 12 attention layers, 12 attention heads and an embeddings size of 768, which the pretrained GPT-2 employs. The model on the other hand does employ an additional BiLSTM encoder during the classification phase, which makes it slower than the unmodified GPT-2 but still faster than the GPT-2 with the BiLSTM + CRF token classification head that employs a computationally demanding CRF layer.

The ablation study clearly shows that the employment of transfer learning is by far the biggest contributor to the overall performance of the system. Surprisingly, there is a very noticeable difference between performances of two distinct pretraining regimes, autoregressive language modelling and masked language modelling in the proposed setting with limited textual resources. Perhaps a masked language modelling objective regime could be somewhat improved by a more sophisticated masking strategy that would not just randomly mask 15 percent of the words but would employ a more fine-grained entity-level masking and phrase-level masking, similar as in Sun *et al.* (2019). This and other pretraining learning objectives will be explored in future work.

In the future we also plan to expand the set of experiments in order to also cover other languages and domains. Since TNT-KID does not require a lot of manually labeled data for fine-tuning and only a relatively small domain specific corpus for pretraining, the system is already fairly transferable to other languages and domains, even to low resource ones. Deploying the system to a morphologically richer language than English and conducting an ablation study in that setting would also allow us to see, whether byte-pair encoding and the additional POS tag sequence input would lead to bigger performance boosts on languages other than English.

Finally, another line of research we plan to investigate is a cross-lingual keyword detection. The idea is to pretrain the model on a multilingual corpus, fine-tune it on one language and then conduct zero-shot cross-lingual testing of the model on the second language. Achieving a satisfactory performance in this setting would make the model transferable even to languages with no manually labeled resources.

Acknowledgements. This paper is supported by European Union's Horizon 2020 research and innovation programme under grant agreement No. 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media). The authors acknowledge also the financial support from the Slovenian Research Agency for research core funding for the programme Knowledge Technologies (No. P2-0103) and the project TermFrame - Terminology and Knowledge Frames across Languages (No. J6-9372).

References

Baevski, Alexei, & Auli, Michael. 2018. Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*.

Beltagy, Iz, Lo, Kyle, & Cohan, Arman. 2019. SciBERT: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Bojanowski, Piotr, Grave, Edouard, Joulin, Armand, & Mikolov, Tomas. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, **5**, 135–146.

Bougouin, Adrien, Boudin, Florian, & Daille, Béatrice. 2013. TopicRank: Graph-based topic ranking for keyphrase extraction. *Pages 543–551 of: Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP)*.

Campos, Ricardo, Mangaravite, Vítor, Pasquali, Arian, Jorge, Alípio Mário, Nunes, Célia, & Jatowt, Adam. 2018a. A Text Feature Based Automatic Keyword Extraction Method for Single Documents. *Pages 684–691 of:* Pasi, Gabriella, Piwowarski, Benjamin, Azzopardi, Leif, & Hanbury, Allan (eds), *Advances in Information Retrieval*. Cham: Springer International Publishing.

Campos, Ricardo, Mangaravite, Vítor, Pasquali, Arian, Jorge, Alípio Mário, Nunes, Célia, & Jatowt, Adam. 2018b. YAKE! collection-independent automatic keyword extractor. *Pages 806–810 of: European Conference on Information Retrieval*. Springer.

Campos, Ricardo, Mangaravite, Vítor, Pasquali, Arian, Jorge, Alípio, Nunes, Célia, & Jatowt, Adam. 2020. YAKE! Keyword extraction from single documents using multiple local features. *Information Sciences*, **509**, 257–289.

Chan, Hou Pong, Chen, Wang, Wang, Lu, & King, Irwin. 2019. Neural keyphrase generation via reinforcement learning with adaptive rewards. *arXiv preprint arXiv:1906.04106*.

Dai, Zihang, Yang, Zhilin, Yang, Yiming, Cohen, William W, Carbonell, Jaime, Le, Quoc V, & Salakhutdinov, Ruslan. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, & Toutanova, Kristina. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

El-Beltagy, Samhaa R, & Rafea, Ahmed. 2009. KP-Miner: A keyphrase extraction system for English and Arabic documents. *Information Systems*, **34**(1), 132–144.

Firoozeh, Nazanin, Nazarenko, Adeline, Alizon, Fabrice, & Daille, Béatrice. 2020. Keyword extraction: Issues and methods. *Natural Language Engineering*, **26**(3), 259–291.

Forney, G David. 1973. The viterbi algorithm. *Proceedings of the IEEE*, **61**(3), 268–278.

Gallina, Ygor, Boudin, Florian, & Daille, Béatrice. 2019. KPTimes: A Large-Scale Dataset for Keyphrase Generation on News Documents. *arXiv preprint arXiv:1911.12559*.

Goldberg, Yoav, & Orwant, Jon. 2013. A dataset of syntactic-ngrams over time from a very large corpus of english books. *Pages 241–247 of: Second Joint Conference on Lexical and Computational Semantics*.

Gollapalli, Sujatha Das, Li, Xiao-Li, & Yang, Peng. 2017. Incorporating expert knowledge into keyphrase extraction. *In: Thirty-First AAAI Conference on Artificial Intelligence*.

Grave, Edouard, Joulin, Armand, Cissé, Moustapha, Jégou, Hervé, *et al.* 2017. Efficient softmax approximation for GPUs. *Pages 1302–1310 of: Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org.

Gu, Jiatao, Lu, Zhengdong, Li, Hang, & Li, Victor OK. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

Hasan, Kazi Saidul, & Ng, Vincent. 2014. Automatic keyphrase extraction: A survey of the state of the art. *Pages 1262–1273 of: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1.

Howard, Jeremy, & Ruder, Sebastian. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Hulth, Anette. 2003. Improved automatic keyword extraction given more linguistic knowledge. *Pages 216–223 of: Proceedings of the 2003 conference on Empirical methods in natural language processing*. Association for Computational Linguistics.

Jawahar, Ganesh, Sagot, Benoît, & Seddah, Djamé. 2019. What does BERT learn about the structure of language?

Kim, Su Nam, Medelyan, Olena, Kan, Min-Yen, & Baldwin, Timothy. 2010. SemEval-2010 Task 5: Automatic Keyphrase Extraction from Scientific Articles. *Pages 21–26 of: Proceedings of the 5th International Workshop on Semantic Evaluation*. SemEval '10. Stroudsburg, PA, USA: Association for Computational Linguistics.

Krapivin, Mikalai, Autaeu, Aliaksandr, & Marchese, Maurizio. 2009. *Large dataset for keyphrases extraction*. Tech. rept. University of Trento.

Kudo, Taku, & Richardson, John. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Lafferty, John, McCallum, Andrew, & Pereira, Fernando CN. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Liu, Yinhan, Ott, Myle, Goyal, Naman, Du, Jingfei, Joshi, Mandar, Chen, Danqi, Levy, Omer, Lewis, Mike, Zettlemoyer, Luke, & Stoyanov, Veselin. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Loper, Edward, & Bird, Steven. 2002. NLTK: The Natural Language Toolkit. *In: Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*.

Luan, Yi, Ostendorf, Mari, & Hajishirzi, Hannaneh. 2017. Scientific information extraction with semi-supervised neural tagging. *arXiv preprint arXiv:1708.06075*.

Medelyan, Olena, Frank, Eibe, & Witten, Ian H. 2009. Human-competitive tagging using automatic keyphrase extraction. *Pages 1318–1327 of: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*. Association for Computational Linguistics.

Meng, Rui, Zhao, Sanqiang, Han, Shuguang, He, Daqing, Brusilovsky, Peter, & Chi, Yu. 2017. Deep keyphrase generation. *arXiv preprint arXiv:1704.06879*.

Meng, Rui, Yuan, Xingdi, Wang, Tong, Brusilovsky, Peter, Trischler, Adam, & He, Daqing. 2019. Does Order Matter? An Empirical Study on Generating Multiple Keyphrases as a Sequence. *arXiv preprint arXiv:1909.03590*.

Mihalcea, Rada, & Tarau, Paul. 2004. Textrank: Bringing order into text. *Pages 404–411 of: Proceedings of the 2004 conference on empirical methods in natural language processing*.

Nguyen, Thuy Dung, & Kan, Min-Yen. 2007. Keyphrase extraction in scientific publications. *Pages 317–326 of: International conference on Asian digital libraries*. Springer.

Paszke, Adam, Gross, Sam, Massa, Francisco, Lerer, Adam, Bradbury, James, Chanan, Gregory, Killeen, Trevor, Lin, Zeming, Gimelshein, Natalia, Antiga, Luca, *et al.* 2019. PyTorch: An imperative style, high-performance deep learning library. *Pages 8024–8035 of: Advances in Neural Information Processing Systems*.

Peters, Matthew E., Neumann, Mark, Iyyer, Mohit, Gardner, Matt, Clark, Christopher, Lee, Kenton, & Zettlemoyer, Luke. 2018a. Deep contextualized word representations. *In: Proc. of NAACL*.

Peters, Matthew E, Neumann, Mark, Iyyer, Mohit, Gardner, Matt, Clark, Christopher, Lee, Kenton, & Zettlemoyer, Luke. 2018b. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Radford, Alec, Wu, Jeffrey, Child, Rewon, Luan, David, Amodei, Dario, & Sutskever, Ilya. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, **1**(8).

Rose, Stuart, Engel, Dave, Cramer, Nick, & Cowley, Wendy. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1–20.

Sahrawat, Dhruva, Mahata, Debanjan, Kulkarni, Mayank, Zhang, Haimin, Gosangi, Rakesh, Stent, Amanda, Sharma, Agniv, Kumar, Yaman, Shah, Rajiv Ratn, & Zimmermann, Roger. 2020. Keyphrase Extraction from Scholarly Articles as Sequence Labeling using Contextualized Embeddings. *Pages 328–335 of: Proceedings of European Conference on Information Retrieval (ECIR 2020)*.

Škrlj, Blaž, Repar, Andraž, & Pollak, Senja. 2019. RaKUn: Rank-based Keyword extraction via Unsupervised learning and Meta vertex aggregation. *Pages 311–323 of: International Conference on Statistical Language and Speech Processing*. Springer.

Sterckx, Lucas, Demeester, Thomas, Deleu, Johannes, & Develder, Chris. 2015. Topical word importance for fast keyphrase extraction. *Pages 121–122 of: Proceedings of the 24th International Conference on World Wide Web*. ACM.

Sun, Yu, Wang, Shuohuan, Li, Yukun, Feng, Shikun, Chen, Xuyi, Zhang, Han, Tian, Xin, Zhu, Danxiang, Tian, Hao, & Wu, Hua. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz, & Polosukhin, Illia. 2017. Attention is all you need. *Pages 5998–6008 of: Advances in neural information processing systems*.

Wan, Xiaojun, & Xiao, Jianguo. 2008. Single Document Keyphrase Extraction Using Neighborhood Knowledge. *Pages 855–860 of: Proceedings of the AAAI Conference*, vol. 8.

Witten, Ian H, Paynter, Gordon W, Frank, Eibe, Gutwin, Carl, & Nevill-Manning, Craig G. 2005. Kea: Practical automated keyphrase extraction. *Pages 129–152 of: Design and Usability of Digital Libraries: Case Studies in the Asia Pacific*. IGI Global.

Yuan, Xingdi, Wang, Tong, Meng, Rui, Thaker, Khushboo, Brusilovsky, Peter, He, Daqing, & Trischler, Adam. 2019. One Size Does Not Fit All: Generating and Evaluating Variable Number of Keyphrases. *arXiv preprint arXiv:1810.05241*.

Appendix: examples of keyword identification

Document 1:

Quantum market games. We propose a quantum-like description of markets and economics. The approach has roots in the recently developed quantum game theory"

Predicted keywords: markets, quantum market games, quantum game theory, economics, quantum like description

True keywords: economics, quantum market games, quantum game theory

Document 2:

Revenue Analysis of a Family of Ranking Rules for Keyword Auctions. Keyword auctions lie at the core of the business models of today's leading search engines. Advertisers bid for placement alongside search results, and are charged for clicks on their ads. Advertisers are typically ranked

according to a score that takes into account their bids and potential clickthrough rates. We consider a family of ranking rules that contains those typically used to model Yahoo! and Google's auction designs as special cases. We find that in general neither of these is necessarily revenue-optimal in equilibrium, and that the choice of ranking rule can be guided by considering the correlation between bidders' values and click-through rates. We propose a simple approach to determine a revenue-optimal ranking rule within our family, taking into account effects on advertiser satisfaction and user experience. We illustrate the approach using Monte-Carlo simulations based on distributions fitted to Yahoo! bid and click-through rate data for a high-volume keyword.

Predicted keywords: ranked, auction, clicks, keyword auctions, keyword, revenue, click-through, ranking rules, click through rates, bids

True keywords: revenue optimal ranking, ranking rule, revenue, advertisement, keyword auction, search engine

Document 3:
Profile-driven instruction level parallel scheduling with application to super blocks. Code scheduling to exploit instruction level parallelism (ILP) is a critical problem in compiler optimization research in light of the increased use of long-instruction-word machines. Unfortunately optimum scheduling is computationally intractable, and one must resort to carefully crafted heuristics in practice. If the scope of application of a scheduling heuristic is limited to basic blocks, considerable performance loss may be incurred at block boundaries. To overcome this obstacle, basic blocks can be coalesced across branches to form larger regions such as super blocks. In the literature, these regions are typically scheduled using algorithms that are either oblivious to profile information (under the assumption that the process of forming the region has fully utilized the profile information), or use the profile information as an addendum to classical scheduling techniques. We believe that even for the simple case of linear code regions such as super blocks, additional performance improvement can be gained by utilizing the profile information in scheduling as well. We propose a general paradigm for converting any profile-insensitive list scheduler to a profile-sensitive scheduler. Our technique is developed via a theoretical analysis of a simplified abstract model of the general problem of profile-driven scheduling over any acyclic code region, yielding a scoring measure for ranking branch instructions.

Predicted keywords: scheduling;instruction level parallel scheduling;instruction level parallelism;profile;list scheduler;code scheduling;long instruction word machines;profile driven scheduling

True keywords: long instruction word machines, scheduling heuristic, compiler optimization, optimum scheduling, abstract model, ranking branch instructions, profile driven instruction level parallel scheduling, profile sensitive scheduler, linear code regions, code scheduling

Document 4:
40 Years After War, Israel Weighs Remaining Risks. JERUSALEM It was 1 p.m. on Saturday, Oct. 6, 1973, the day of Yom Kippur, the holiest in the Jewish calendar, and Israel's military intelligence chief, Maj. Gen. Eli Zeira, had called in the country's top military journalists for an urgent briefing. He told us that war would break out at sundown, about 6 p.m., said Nachman Shai, who was then the military affairs correspondent for Israel's public television channel and is now a Labor member of Parliament. Forty minutes later he was handed a note and said, Gentlemen, the war broke out, and he left the room. Moments before that note arrived, according to someone else who was at that meeting, General Zeira had been carefully peeling almonds in a bowl of ice water. The coordinated attack by Egypt and Syria, which were bent on regaining strategic territories and pride

lost to Israel in the 1967 war, surprised and traumatized Israel. For months, its leaders misread the signals and wrongly assumed that Israel's enemies were not ready to attack. Even in those final hours, when the signs were unmistakable that a conflict was imminent, Israel was misled by false intelligence about when it would start. As the country's military hurriedly called up its reserves and struggled for days to contain, then repel, the joint assault, a sense of doom spread through the country. Many feared a catastrophe. Forty years later, Israel is again marking Yom Kippur, which falls on Saturday, the anniversary of the 1973 war according to the Hebrew calendar. This year the holy day comes in the shadow of new regional tensions and a decision by the United States to opt, at least for now, for a diplomatic agreement rather than a military strike against Syria in response to a deadly chemical weapons attack in the Damascus suburbs on Aug. 21. Israeli newspapers and television and radio programs have been filled with recollections of the 1973 war, even as the country's leaders have insisted that the probability of any new Israeli entanglement remains low and that the population should carry on as normal. For some people here, though, the echoes of the past have stirred latent questions about the reliability of intelligence assessments and the risks of another surprise attack. Any Israeli with a 40-year perspective will have doubts,said Mr. Shai, who was the military's chief spokesman during the Persian Gulf War of 1991, when Israelis huddled in sealed rooms and donned gas masks, shocked once again as Iraqi Scud missiles slammed into the heart of Tel Aviv. Coming after the euphoria of Israel's victory in the 1967 war, when six days of fighting against the Egyptian, Jordanian and Syrian Armies left Israel in control of the Sinai Peninsula, the West Bank, Gaza, East Jerusalem and the Golan Heights, the conflicts of 1973, 1991 and later years have scarred the national psyche. But several former security officials and analysts said that while the risks now may be similar to those of past years in some respects, there are also major differences. In 1991, for example, the United States responded to the Iraqi attack by hastily redeploying some Patriot antimissile batteries to Israel from Europe, but the batteries failed to intercept a single Iraqi Scud, tracking them instead and following them to the ground with a thud. Since then, Israel and the United States have invested billions of dollars in Israel's air defenses, with the Arrow, Patriot and Iron Dome systems now honed to intercept short-, medium- and longer-range rockets and missiles. Israelis, conditioned by subsequent conflicts with Hezbollah in Lebanon and Hamas in Gaza and by numerous domestic drills, have become accustomed to the wail of sirens and the idea of rocket attacks. But the country is less prepared for a major chemical attack, even though chemical weapons were used across its northern frontier, in Syria, less than a month ago, which led to a run on gas masks at distribution centers here. In what some people see as a new sign of government complacency at best and downright failure at worst, officials say there are enough protective kits for only 60 percent of the population, and supplies are dwindling fast. Israeli security assessments rate the probability of any attack on Israel as low, and the chances of a chemical attack as next to zero. In 1973, the failure of intelligence assessments about Egypt and Syria was twofold. They misjudged the countries' intentions and miscalculated their military capabilities. Our coverage of human intelligence, signals intelligence and other sorts was second to none, said Efraim Halevy, a former chief of Mossad, Israel's national intelligence agency. We thought we could initially contain any attack or repulse it within a couple of days. We wrongly assessed the capabilities of the Egyptians and the Syrians. In my opinion, that was the crucial failure. Israel is in a different situation today, Mr. Halevy said. The Syrian armed forces are depleted and focused on fighting their domestic battles, he said. The Egyptian Army is busy dealing with its internal turmoil, including a campaign against Islamic militants in Sinai. Hezbollah, the Lebanese militant group, is heavily involved in aiding President Bashar al-Assad of Syria, while the Iranians, Mr. Halevy said, are not likely to want to give Israel a reason to strike them, not as the aggressor but as a victim of an Iranian attack. Israel is also much less likely to suffer such a colossal failure in assessment, Mr. Halevy said. We have plurality in the intelligence community, and people have learned to speak up, he said. The danger of a mistaken concept is still there, because we are human. But it is much more remote than before. Many analysts have attributed the failure of 1973 to arrogance. There was a disregarding of intelligence, said Shlomo

Avineri, a political scientist at Hebrew University and a director general of Israel's Ministry of Foreign Affairs in the mid-1970s. War is a maximization of uncertainties, he said, adding that things never happen the same way twice, and that wars never end the way they are expected to. Like most countries, Israel has been surprised by many events in recent years. The two Palestinian uprisings broke out unexpectedly, as did the Arab Spring and the two revolutions in Egypt. In 1973, logic said that Egypt and Syria would not attack, and for good reasons, said Ephraim Kam, a strategic intelligence expert at the Institute for National Security Studies at Tel Aviv University who served for more than 20 years in military intelligence. But there are always things we do not know. Intelligence is always partial, Mr. Kam said, its gaps filled by logic and assessment. The problem, he said, is that you cannot guarantee that the logic will fit with reality. In his recently published diaries from 1973, Uzi Eilam, a retired general, recalled the sounding of sirens at 2 p.m. on Yom Kippur and his rushing to the war headquarters. Eli Zeira passed me, pale-faced, he wrote, referring to the military intelligence chief, and he said: So it is starting after all. They are putting up planes. A fleeting glance told me that this was no longer the Eli Zeira who was so self-assured.

Predicted keywords: israel, syria, egypt, military, jerusalem

True keywords: israel, yom kippur, egypt, syria, military, arab spring

Document 5:
Abe's 15-month reversal budget fudges cost of swapping people and butter for concrete and guns. The government of Shinzo Abe has just unveiled its budget for fiscal 2013 starting in April. Abe's stated intention was to radically reset spending priorities. He is indeed a man of his word. For this is a budget that is truly awesome for its radical step backward into the past a past where every public spending project would do wonders to boost economic growth. It is also a past where a cheaper yen would bring unmitigated benefits to Japan's exporting industries. None of it is really true anymore. Public works do indeed do wonders in boosting growth when there is nothing there to begin with. But in a mature and well-developed economy like ours, which is already so well equipped with all the necessities of modern life, they can at best have only a one-off effect in creating jobs and demand. And in this globalized day and age, an exporting industry imports almost as much as it exports. No longer do we live in a world where a carmaker makes everything within the borderlines of its nationality. Abe's radical reset has just as much to do with philosophy as with timelines. Three phrases come to mind as I try to put this budget in a nutshell. They are: from people to concrete,from the regions to the center and from butter to guns. The previous government led by the Democratic Party of Japan declared that it would put people before concrete. No more building of ever-empty concert halls and useless multiple amenity centers where nothing ever happens. More money would be spent on helping people escape their economic difficulties. They would give more power to the regions so they could decide for themselves what was really good and worked for the local community. Guns would most certainly not take precedence over butter. Or rather over the low-fat butter alternatives popular in these more health-conscious times. All of this has been completely reversed in Abe's fiscal 2013 budget. Public works spending is scheduled to go up by more than 15 percent while subsistence payments for people on welfare will be thrashed to the tune of more than 7 percent. If implemented, this will be the largest cut ever in welfare assistance. The previous government set aside a lump sum to be transferred from the central government's coffers to regional municipalities to be spent at their own discretion on local projects. This sum will now be clawed back into the central government's own public works program. The planned increase in spending on guns is admittedly small: a 0.8 percent increase over the fiscal 2012 initial budget. It is nonetheless the first increase of its kind in 11 years. And given the thrashing being dealt to welfare spending, the shift in emphasis from butter to guns is clearly apparent. One of the Abe government's boasts is that it will manage to hold down the overall size of the budget in comparison with fiscal 2012. The other one is that it will raise more revenues

from taxes rather than borrowing. True enough on the face of it. But one has to remember the very big supplementary budget that the government intends to push through for the remainder of fiscal 2012. The money for that program will come mostly from borrowing. Since the government is talking about a 15-month budget that seamlessly links up the fiscal 2012 supplementary and fiscal 2013 initial budgets, they should talk in the same vein about the size of their spending and the borrowing needed to accommodate the whole 15-month package. It will not do to smother the big reset with a big coverup.

Predicted keywords: shinzo abe, japan, budget

True keywords: shinzo abe, budget

Appendix B: Bisociative Literature-Based Discovery: Lessons Learned and New Word Embedding Approach

New Generation Computing (2020) 38:773–800
https://doi.org/10.1007/s00354-020-00108-w

Bisociative Literature-Based Discovery: Lessons Learned and New Word Embedding Approach

Nada Lavrač[1,2] · Matej Martinc[1,3] · Senja Pollak[1] · Maruša Pompe Novak[4] · Bojan Cestnik[1,5]

Received: 21 March 2020 / Accepted: 8 September 2020 / Published online: 6 October 2020

Abstract
The field of bisociative literature-based discovery aims at mining scientific literature to reveal yet uncovered connections between different fields of specialization. This paper outlines several outlier-based literature mining approaches to bridging term detection and the lessons learned from selected biomedical literature-based discovery applications. The paper addresses also new prospects in bisociative literature-based discovery, proposing an advanced embeddings-based technology for cross-domain literature mining.

Keywords Literature-based discovery · Cross-domain bisociations · Computational creativity · Embeddings technology

✉ Bojan Cestnik
bojan.cestnik@temida.si

Nada Lavrač
nada.lavrac@ijs.si

Matej Martinc
matej.martinc@ijs.si

Senja Pollak
senja.pollak@ijs.si

Maruša Pompe Novak
Marusa.Pompe.Novak@nib.si

[1] Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia

[2] University of Nova Gorica, Vipavska 13, 5000 Nova Gorica, Slovenia

[3] Jožef Stefan International Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia

[4] National Institute of Biology, Večna pot 111, 1000 Ljubljana, Slovenia

[5] Temida d.o.o., Dunajska cesta 51, 1000 Ljubljana, Slovenia

Introduction

Growing amounts of available knowledge and data exceed human analytic capabilities. Therefore, new technologies that help analyzing and extracting useful information from large amounts of data need to be developed and used for analytic purposes. Understanding complex phenomena and solving difficult problems often require knowledge from different domains to be combined and cross-domain associations to be considered. While the concept of association is at the heart of several information technologies, including information retrieval and data mining, and in particular association rule learning [2], scientific discovery requires creative thinking to connect seemingly unrelated information, for example, using metaphors or analogies between concepts from different domains. These kinds of context crossing associations, called bisociations [19], are often needed for innovative discoveries.

This paper addresses a computational creativity task of bisociative knowledge discovery from scientific literature that we name bisociative literature-based discovery. This task is at the intersection of two research areas: literature-based discovery [6] and bisociative knowledge discovery [3], which are briefly introduced below.

In literature-based discovery (LBD) [6]—and in particular in cross-domain literature mining that addresses knowledge discovery from two (or more) initially separate document corpora—a crucial step is the identification of interesting bridging terms (b-terms) or links (b-links) that carry the potential of explicitly revealing the connections between the separate domains. Swanson and Smalheiser [37, 40] developed early LBD approaches to detecting interesting b-terms to uncover the possible cross-domain relations among previously unrelated concepts. Their approach, known as the 'ABC model of knowledge discovery', addresses the so-called closed discoverysetting [43], where two initially separate domains A and C are specified by the user at the beginning of the discovery process, and the goal is to search for bridging concepts (b-terms) in B to validate the hypothesized connection between A and C.

Similarly, bisociative knowledge discovery [3] addresses a data mining task where two (or more) domains of interest are searched for bridging concepts (bridging terms or links). Using either the same representation of different domains or different representations of the same domain, bridging concepts can be detected either as nodes bridging different graphs, as subgraphs linking different graphs, as bridging links in terms of graph similarity, or as bridging terms appearing in separate document corpora, which is referred to as bridging term discovery in this paper.

Until recently, literature-based discovery and bisociative knowledge discovery approaches to cross-domain literature mining used conventional bag of words (BoW) vector representation of text, using term frequency inverse document frequency (TF-IDF) word weighting heuristics. Recent text-mining approaches started exploiting neural networks-based text representations, using text embedding methods that use large corpora of documents to extract numeric vector

representations for words, sentences, and/or documents. In this paper, we exploit the power of word embeddings [25, 27], which refer to vector representations of words, where each word is assigned a vector of several hundred dimensions in the transformed n-dimensional numeric vector space. Embedding approaches have started emerging also in the area of computational creativity [1, 10] and literature-based discovery [24].

The contributions of this paper are many-fold. The paper first reflects on the lessons learned from our past research in cross-domain literature mining,[1] focusing on outlier document detection as means for more effectively searching for novel bridging terms. Second, we propose an embedding-inspired conceptual framework for creative bisociative LBD, based on a novel concept of bridging by relational bisociation. Third, we propose a new bisociative LBD methodology, using word embeddings for relational bisociation discovery. Finally, we show-case the potential utility of this approach on a new biological research problem of finding connections between circadian rhythm and plant defense domains, where the results of this proof of concept evaluation indicate that the new methodology is very relevant for LBD research.

The paper is structured as follows. "Background and related work" presents the related work in literature-based discovery (LBD) and bisociative knowledge discovery, including the previously published relationship between the two [21, 31]. It presents also the related work in representation learning using the embedding technology. "Past LBD results and lessons learned" outlines selected approaches to cross-domain literature mining via outlier document detection and exploration [31, 36], together with the lessons learned from this research. "Towards creative embeddings-based bisociative LBD" proposes a novel creative discovery research direction based on the recent word embedding technology, with a proof of concept experiment in a biological domain, together with the lessons learned from this LBD application. Finally, "Conclusions and further work" concludes with a summary and plans for further research.

Background and Related Work

This section presents the related work. "Literature-based discovery" introduces literature-based discovery (LBD), which is the main topic of this research. "Bisociative knowledge discovery" presents the area of computational creativity named bisociative knowledge discovery and the connection between bisociative knowledge discovery and LBD, as published in our past research [21, 31]. Finally, "Embeddings" briefly introduces embeddings, the contemporary representation learning technology resulting from recent research in neural networks, which is the enabler for the proposed embedding-based bisociative LBD methodology introduced in "Towards creative embeddings-based bisociative LBD".

[1] These lessons have been published also in the ICCC-2020 paper by Lavrač et al. [22].

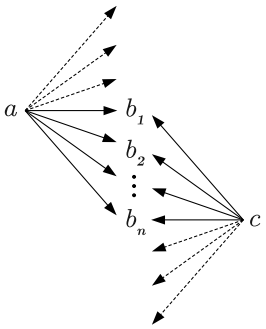


Fig. 1 Closed discovery process defined by Weeber et al. [43]

Literature-Based Discovery

In literature-based discovery (LBD) [6]—and in particular in cross-domain literature mining, which addresses knowledge discovery in two (several) initially separate document corpora—a crucial step is the identification of interesting bridging terms (b-terms) that carry the potential of revealing the links connecting the separate domains.

Early work in LBD [37, 40] developed approaches to assist the user in literature-based discovery by detecting interesting cross-domain terms with a goal to uncover the possible relations between previously unrelated concepts. The ARROWSMITH online system, developed by Smalheiser and Swanson [37], takes as input two sets of titles of scientific papers from disjoint domains (disjoint document corpora) A and C, and lists terms that are common to A and C; the resulting bridging terms (b-terms) are further investigated by the user for their potential to generate new scientific hypotheses.[2] Their approach, known as the 'ABC model of knowledge discovery', addresses several settings, including the closed discovery setting [43], where two initially separate domains A and C are specified by the user at the beginning of the discovery process, and the goal is to search for bridging concept (term) b in B to support the validation of the hypothesized connection between A and C. The closed discovery setting, which is the most frequently addressed LBD setting, is illustrated in Fig. 1.

Swanson's seminal work has shown that databases such as PubMed can serve as a rich source of yet hidden relations between usually unrelated topics, potentially leading to novel insights and discoveries. By studying two separate literatures, i.e., the literature on migraine headache and the articles on magnesium, Swanson [39] discovered 'Eleven neglected connections', all of them supportive for the hypothesis that magnesium deficiency might cause migraine headache. Figure 2 illustrates the closed discovery setting on the Swanson's task of finding the terms linking the 'migraine' and 'magnesium' domains. Swanson's literature mining results have been later confirmed by laboratory and clinical investigations. This well-known example

[2] In the ABC model, uppercase letter symbols A, B, and C are used to represent concepts (or sets of terms), and lowercase symbols a, b, and c to represent single terms.

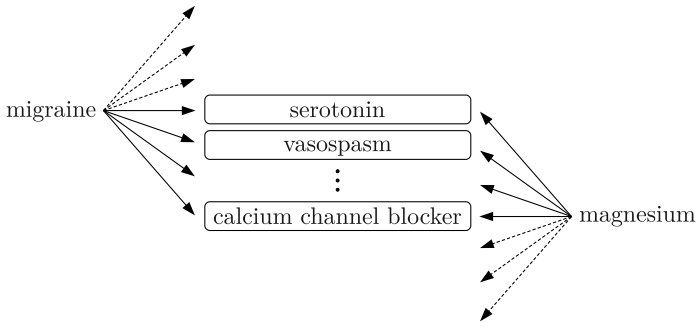


Fig. 2 Closed discovery when exploring migraine and magnesium documents, with b-terms identified by Swanson et al. [41]

has become the gold standard in the literature mining field and has been used as a benchmark in several studies [17, 23, 38, 43].

Inspired by this early work, literature mining approaches were further developed and successfully applied to different problems, such as finding associations between genes and diseases [16], diseases and chemicals [44], and others. Supporting the user in effectively searching for bridging terms (b-terms) provided a motivation for developing the CrossBee approach to bridging term detection applicable in the closed discovery setting [17], implemented through ensemble-based term ranking, where an ensemble heuristic composed of six elementary heuristics was constructed for term evaluation.

The work of Kastrin et al. [18] is complementary to other LBD approaches, as it uses different similarity measures (such as common neighbors, Jaccard index, and preferential attachment) for link prediction of implicit relationships in the Semantic MEDLINE network. Holzinger et al. [15] describe several web-based tools for the analysis of biomedical literature, which include the analysis of terms (biomedical entities such as disease, drugs, genes, proteins, and organs) and provide concepts associated with a given term. A comprehensive survey of modern literature-based discovery approaches in biomedical domain can be found in [13, 33].

Our past research [31, 36] suggests that bridging terms are more frequent in documents that are in some sense different from the majority of documents in a given domain. For example, Sluban et al. [36] have shown that such documents, considered being outlier documents of their own domain, contain a substantially larger amount of bridging/linking terms than the regular non-outlier documents. This approach, using the OntoGen tool [12], is described in some detail in "Past LBD results and lessons learned".

Bisociative Knowledge Discovery

Bisociative knowledge discovery is a challenging task motivated by a trend of over-specialization in the research and development, which usually results in deep and relatively isolated silos of knowledge. Scientific literature too often remains closed and cited only in professional subcommunities. The information that is related across different contexts is difficult to identify using associative approaches, like

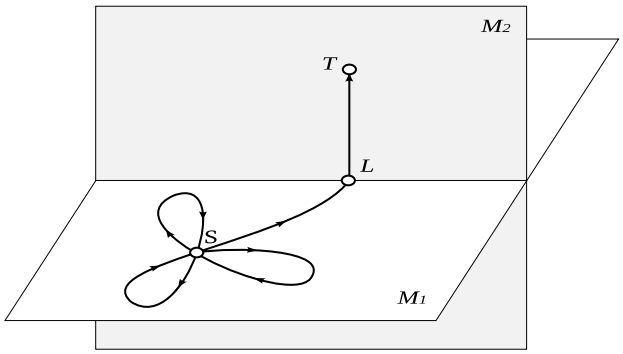


Fig. 3 Koestler's schema of bisociative discovery in science [19, p. 107], illustrating the creative act of finding links (from *S* to target *T*) that lead 'out-of-the-plane' via intermediate, bridging concepts (*L*)

the standard association rule learning [2] known from the data mining and machine learning literature. Therefore, the ability of literature mining methods and software tools to support the experts in their knowledge discovery processes—especially in searching for yet unexplored connections between different domains—is becoming increasingly important.

Arthur Koestler [19] argued that the essence of creativity lies in "perceiving of a situation or idea . . . in two self-consistent but habitually incompatible frames of reference", and introduced the expression bisociation to characterize this creative act. More specifically, Koestler's notion of bisociation was originally defined as follows.

> "The pattern ... is the perceiving of a situation or idea, *L*, in two self-consistent but habitually incompatible frames of reference, M_1 and M_2. The event *L*, in which the two intersect, is made to vibrate simultaneously on two different wavelengths, as it were. While this unusual situation lasts, *L* is not merely linked to one associative context but bisociated with two."

Koestler found bisociation to be the basis for human creativity in seemingly diverse human endeavors, such as humor, science, and arts. The concept of bisociation is illustrated in Fig. 3. It should be noted that context crossing is subjective, since the user has to move from his 'normal' context (frame of reference) to an *habitually incompatible context* to find the bisociative link. In Koestler's terms (Fig. 3), a habitual frame of reference (plane M_1) corresponds to the domain defined by the user. Other domains represents different, habitually incompatible contexts (in general, there may be several planes M_2), where the creative act is to find links that lead 'out-of-the-plane' via intermediate, bridging concepts. Thus, contextualization and link discovery are two of the fundamental mechanisms in bisociative reasoning.

In summary, according to Koestler [19], bisociative thinking occurs when a problem, idea, event, or situation is perceived simultaneously in two or more 'matrices of thought' or domains. When two matrices of thought interact with each other, the result is either their fusion in a novel intellectual synthesis or their confrontation in a new aesthetic experience. Koestler regarded many different mental phenomena that are based on comparison (such as analogies, metaphors, jokes, identification, and anthropomorphism) as special cases of bisociation.

More recently, this work was followed by the researchers interested in the so-called bisociative knowledge discovery, where—according to [3]—two concepts are

Table 1 Unifying Koestler's and Swanson's models of creative knowledge discovery [21, 31]

Koestler's model	Swanson's model
Bisociative link discovery process	Closed discovery process
Frames of reference (contexts) M_1 and M_2	Domains of interest A and C
Bisociative cross-context link $L \in M_1 \cap M_2$	Bridging term $b \in$ terms$(A) \cap$ terms (C)

bisociated if there is no direct, obvious evidence linking them and if one has to cross different domains to find the link, where a new link must provide some novel insight into the problem addressed. Bisociative knowledge discovery has become a topic of extensive research, addressing the discovery of bridging links or bridging concepts crossing between different domains and representations.

In conclusion, let us summarize the previously published [21, 31] relationship between bisociative knowledge discovery and Swanson's ABC model for literature-based discovery, where the particular focus of interest is the relationship between Koestler's bisociative link discovery framework and Weeber's closed discovery framework, as summarized in Table 1. Similar to a bisociation, which is according to Koestler a result of processes of mind when making new associations between concepts S and T from usually separated contexts (illustrated in Fig. 3), literature-based discoveries in Swanson's ABC model are a result of uncovering links between concepts a and c from disjoint literatures A and C (illustrated in Fig. 1). In terms of Koestler's model, the two domains A and C, investigated in the closed literature-based discovery framework, correspond to the two habitually incompatible frames of reference, M_1 and M_2. Moreover, the bridging terms $b_1, b_2, \ldots, b_n$ that are common to literature A and C clearly correspond to Koestler's notion of a situation or idea, L, which is not merely linked to one associative context, but bisociated with two contexts M_1 and M_2.

Embeddings

In terms of representation learning, our past LBD research that led to the lessons learned described in "Past LBD results and lessons learned" was based on using the standard TF-IDF weighted BoW vector representations of text documents [7, 17, 31, 36]. On the other hand, the novel LBD methodology proposed in this paper in "Towards creative embeddings-based bisociative LBD" exploits contemporary representations of text documents using embeddings, given that current research in natural language processing demonstrates that representation learning using embeddings is much more effective than using the standard TF-IDF BoW vector representation. The embedding approach to representation learning can be defined as follows.

Embeddings Given input data of a given data type and format, find a tabular representation of the data, where each row represents a single data

instance, and each column represents one of the dimensions in the d-dimensional numeric vector space $\mathbb{R}^d$.

The embedding technology is a prominent side effect of the recent revival of neural networks (NN), in which the information is represented by activation patterns in interconnected networks of primitive units (neurons). This enables concepts to be gradually learned by an NN from the observed data by modifying the connection weights between the hierarchically organized units. These weights that can be extracted from neural networks can be used as a spatial representation that transforms relations between observed entities (data instances) into distances.

Recently, the embedding approach became a prevalent way to build representations for many different types of entities, e.g., graphs, electronic health records, images, relations, recommendations, as well as texts (documents, sentences and/or words). Word embeddings [25, 27], which are in the focus of our research described in "Towards creative embeddings-based bisociative LBD", use large corpora of documents to extract vector representations of words, assigning each word a vector of several hundred dimensions. The first neural word embeddings like word2vec [25] produced one vector for each word, irrespective of its polysemy (e.g., for a polysemous word like bank, word2vec produces a single representation vector, and ignores the fact that bank can present both a financial institution and a land sloping down to a water mass). Recent developments like ELMo [30] and BERT [9] take a context of a sentence into account and produce different word vectors for different contexts of each word. A further improvement of neural word embeddings for texts uses multitask prediction (inclusion of several related textual prediction tasks).

Past LBD Results and Lessons Learned

Outliers, characterized by their properties of being infrequent or unusual, may represent unexpected events, entities, items, or documents. Early research in LBD has focused on the identification and exploration of outlier documents, since they frequently embody new information that is often hard to explain in the context of existing mainstream knowledge. The LBD research by Petrič et al. [31] and Sluban et al. [36] suggests that bridging terms are more frequent in documents that are in some sense different from the majority of documents in a given domain.

The outlier-based approach to LBD proposed by Petrič et al. [31] uses document clustering to find outlier documents. The approach consists of two steps. In the first step, the OntoGen clustering algorithm by Fortuna et al. [12] is applied to cluster the merged document set A ∪ C, consisting of documents from two domains A and C. The result of unsupervised clustering is two document clusters: A′ = Classified as *A* (i.e., documents from A ∪ C classified as A), and C′= Classified as *C* (i.e., documents from A ∪ C classified as C). In the second step of outlier detection, clusters A′ and C′ are further separated, each into two clusters, based on the documents' original labels A and C. As a result, a two-level tree hierarchy of clusters is generated, as illustrated in Fig. 4.

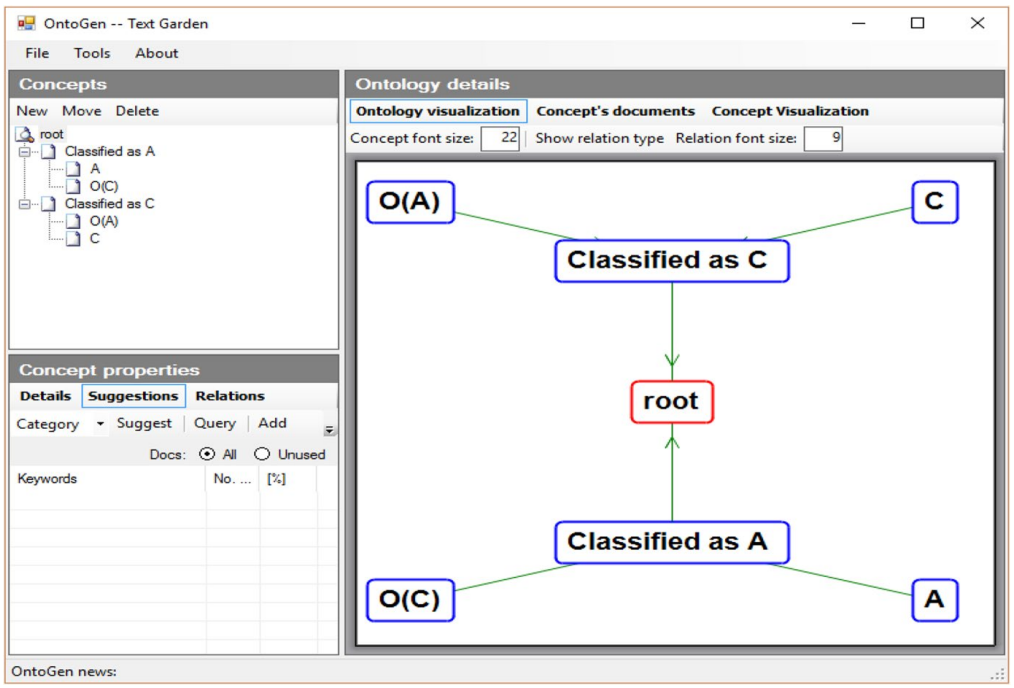


Fig. 4 Target domain documents from literatures A and C, clustered according to the OntoGen's two-step approach, first using unsupervised and then supervised clustering to obtain outlier documents $O(A)$ and $O(C)$ of literatures A and C, respectively. The figure illustrates the outlier detection approach implemented using OntoGen, addressing the outlier detection framework that is conceptually explained in Fig. 5

Lesson Learned 1: Potential of outlier documents

The hypothesis that outlier documents have the potential to improve the effectiveness of bridging term detection was tested on the migraine–magnesium [41] and autism–calcineurin [32] domain pair datasets, which have lists of concept bridging terms (b-terms) confirmed by the medical experts. The experimental results obtained using OntoGen confirm the hypothesis that most bridging terms appear in outlier documents and that by considering only outlier documents, the search space for b-term identification can be largely reduced.

This lesson—that outlier documents have the potential for improving the effectiveness of bridging term detection—was reconfirmed in the work of Sluban et al. [36], exploring a classification filtering approach to outlier detection, which was tested on the same domain pair data sets, migraine–magnesium [41] and autism–calcineurin [32] domain, which have lists of bridging terms (b-terms) confirmed by the medical experts. Sluban et al. [36] proposed to detect outlier documents using

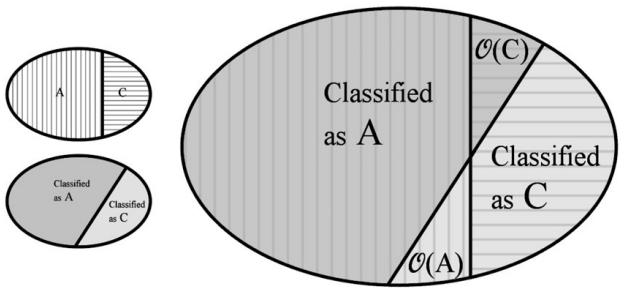


Fig. 5 Detecting outliers of a domain pair dataset $A \cup C$, using a document classification approach by Sluban et al. [36]

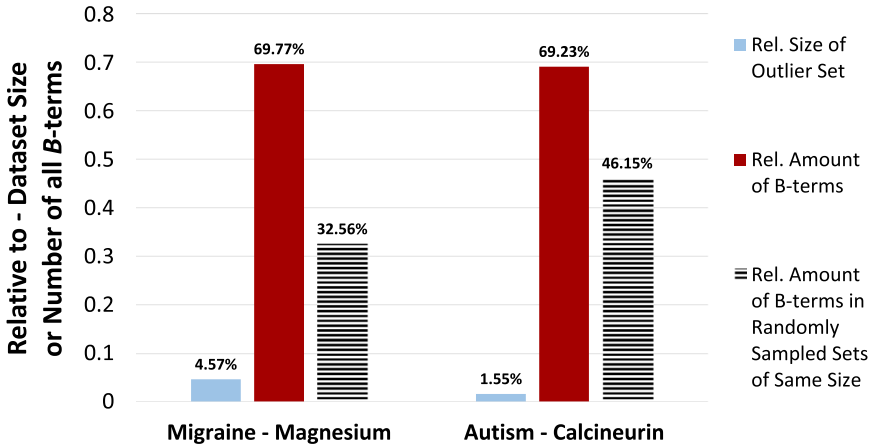


Fig. 6 Presence of b-terms in the detected outlier sets of two domain pair datasets

classification algorithms for classification noise filtering, first suggested by Brodley and Friedl [5]. Having documents from two domains of interest A and C, Sluban et al. [36] first trained an ensemble classifier that distinguishes between the documents of these domains, and use the ensemble classifier to classify all the documents. The miss-classified documents were declared as outliers, since—according to the classification model—they do not belong to their domain (class label) of origin. These outliers can be interpreted as borderline documents as they were considered by the model to be more similar to the other domain than to their original domain, and can be regarded as bridging documents between the two domains. In other words, if an instance of class A is classified in the opposite class C, it is considered an outlier of domain A, and vice versa. The two sets of outlier documents were denoted with $O(A)$ and $O(C)$, as illustrated in Fig. 5.

The experimental results obtained by Sluban et al. [36] showed that the sets of detected outlier documents are relatively small—including less than 5% of the entire datasets—and that they contain a great majority of bridging terms previously identified by medical experts, which was significantly higher than in same-sized random document subsets. These results are summarized in Fig. 6.

These experimental results indicate that it is justified that the search for b-terms can be focused on outlier documents, which contain a large majority of b-terms. Consequently, by focusing the exploration on outlier documents, the effort needed for finding cross-domain links is substantially reduced, as it requires to explore a

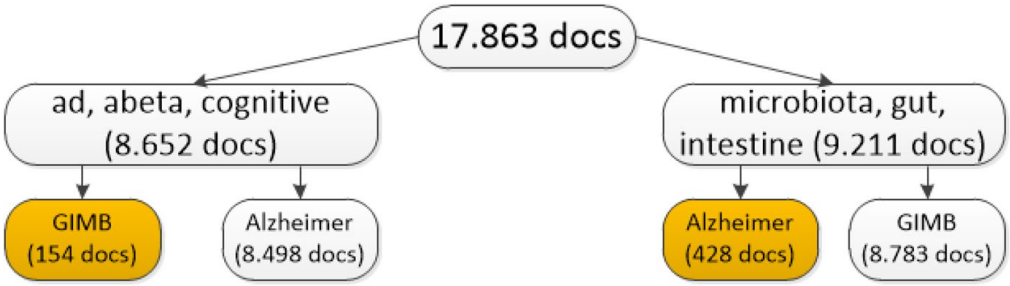


Fig. 7 Two-level cluster hierarchy constructed with ontoGen from the dataset of 17,863 papers in the Alzheimer's disease–gut microbiome domain pair

much smaller subset of documents, where a great majority of b-terms are present and more frequent.

When applying OntoGen on the documents of the new application domain using the Alzheimer's disease–gut microbiome domain pair [7], the OntoGen method uses domains A and C, and builds a joint document set $A \cup C$. With this intention, two individual sets of documents (e.g., titles, abstracts, or full texts of scientific articles), one for each domain under research (namely, literature A on Alzheimer's disease and literature C on gut microbiome), were automatically retrieved from the PubMed database. A cluster hierarchy was constructed from the dataset of 17,863 papers with OntoGen. Two first-level clusters are labeled with the OntoGen suggested keywords ad, abeta, cognitive, and microbiota, gut, and intestine. Four second-level subclusters separate documents according to their original search keywords for Alzheimer's disease and gut microbiome, as illustrated in Fig. 7.

Lesson Learned 2: Excluding intersecting documents

In Alzheimer's disease–gut microbiome LBD application, the initial document set $A \cup C$ consisted of some documents, which were in the intersection of A and C, meaning that a few documents were retrieved from PubMed by both of the two separate queries for domain A (i.e., Alzheimer and C (i.e., (gut OR intestinal) AND (microbiota OR bacteria)), which was surprising. After carefully inspecting these documents (as these documents could contain the b-terms representing a solution to the problem, which proved not to be the case), it was realized that keeping them in the $A \cup C$ document set was problematic. As a result, the documents that were retrieved by both queries were eliminated,[3]

[3] Their inclusion in the document set would have violated the assumption of literature-based discovery and bisociative knowledge discovery frameworks, which assume that the explored literature domains A and C are disjoint; if this assumption was violated, the methodology would fail due to biased heuristics calculations.

resulting in 17,863 documents kept in the $A \cup C$ document set used for further exploration.

Lesson Learned 3: Selecting only outlier documents

The hypothesis that the search for bridging terms can be reduced to manageable subsets of documents was confirmed in our experiments. In the Alzheimer's disease–gut microbiome LBD application using OntoGen for outlier document detection, the space of documents used for b-term exploration was further reduced from the set of 17,863 documents to two subsets of outlier documents, i.e., to only 154 gut microbiome papers and 428 Alzheimer's disease related papers, considered as outliers in their own domain, leading to the selection of only 582 documents for further inspection.

Lesson Learned 4: Expert revision of b-terms list

The hypothesis that b-terms selected from outlier documents can be further reduced with expert knowledge was confirmed in our experiments. By processing the remaining 582 outlier documents, we used CrossBee [17] to extract 4723 terms as potential b-terms connecting the two domains. In b-term exploration, all the terms were considered and not just the medical ones, except that a list of 523 English stop words was used to filter out meaningless words, and English Porter stemming was applied. Even though the list of potential bridging terms was ordered according to the ensemble-heuristics estimated bridging terms potential, browsing and analyzing the terms from the list still presented a substantial burden for the domain expert. To further reduce the size of the potential b-term list, the collaborating domain expert[4] prepared a list of 289 domain terms of her own research interest. This list included common terms and specific molecular factors and pathways, which were manually identified in titles, abstracts, and keywords from 42 papers obtained from PubMed search query (gut AND Alzheimer), 55 of which appeared also among the 4723 terms extracted by CrossBee. During the evaluation phase, the relevant papers for each b-term candidate were reviewed and searched for potential clues justifying further investigation, resulting

[4] Elsa Fabretti.

from relevant b-term discoveries confirmed by the domain expert [7].

Compared to outlier document detection using OntoGen, an upgraded methodology proposed by Cestnik et al. [7] was implemented in a reusable outlier-based LBD methodology in a web-based text-mining platform TextFlows[5] [29] that allowed us to construct and execute advanced text-mining workflows. The workflow shown in Fig. 8 consists of seven steps implemented as subprocesses. The connections between subprocesses represent the flow of documents from one subprocess to another. In overview, steps 1–3 represent the outlier detection part, and steps 4–7 represent cross-domain exploration for b-term detection.

Lesson Learned 5: TextFlows workflow helping experts

In the experiments using the TextFlows workflow, the NoiseRank ensemble-based outlier detection approach [35] implemented in TextFlows was used. The goal of the first three steps (using first three workflow widgets) of the methodology is to effectively extract a set of outlier documents from the whole corpus of input documents. Consequently, by decreasing the size of the input set of documents, the second phase becomes more focused, efficient, and effective. In the last four steps of the workflow in Fig. 8, components that constitute the CrossBee HCI interface [17] are executed to conduct expert-guided b-term analysis. Here, the goal is to further prepare the input documents for b-term visualization and exploration. Note that in this step, the role of the domain expert is crucial.

Towards Creative Embeddings-Based Bisociative LBD

In this section, we first formally define bisociation and the specific bisociative patterns that are searched for bisociative knowledge discovery (i.e., bridging concepts, bridging graphs, and bridging by structural similarity), including the novel concept of bridging by relational bisociation in "Formal framework for creative bisociative LBD". The potential of the embeddings technology for creative knowledge discovery is explained in "Word embeddings potential for creative knowledge discovery". "Novel embeddings-based bisociative LBD methodology" presents the proposed word embeddings-based bisociative LBD methodology, and explores the creativity

[5] http://textflows.org.

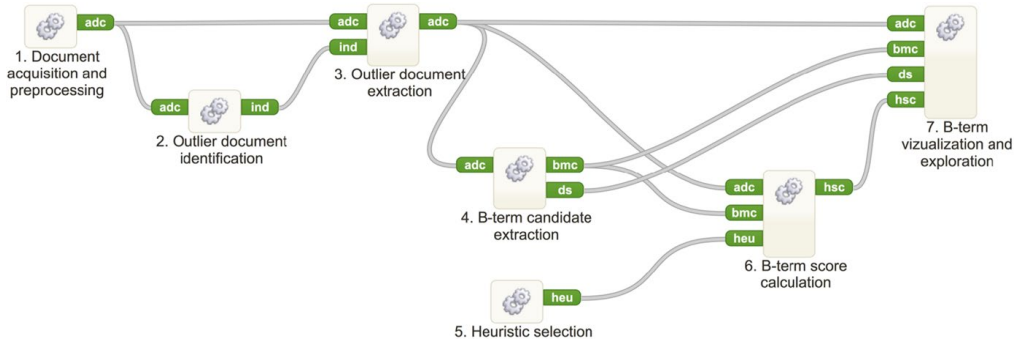


Fig. 8 A top-level workflow of the LBD methodology in TextFlows [29]

potential of word embeddings in an LBD closed discovery setting, assuming an expert-defined relationship of interest between two terms a_1 and a_2 in domain A and an unknown relationship to be discovered for a given seed concept c and an unknown/yet to be discovered term x in domain C. "Embeddings-based relational LBD experiment conducted on thecircadian rhythm and plant defense domains" briefly outlines the experimental setting of the experiments conducted on the circadian rhythm–plant defense domain pair, where a proof-of-concept result evaluation is given in "Results". This section concludes by a summary and lessons learned from these experiments in "Summary and lesson learned from these experiments".

Formal Framework for Creative Bisociative LBD

Bisociation is essentially a creative endeavor. To connect pieces of information from previously unrelated domains, a person must activate some form of creative mechanism. This creative aspect is what allows one to go beyond one-dimensional associations. This has been recognized in several psycho-cognitive theories related to creativity, which share the principle that a strong connection exists between creative activity and the ability to establish relations between seemingly unrelated domains.

Divergent reasoning can be achieved—to a certain degree—by means of cross-domain exploration in multi-domain databases. Such a model must provide mechanisms for mapping concepts and transferring meanings. According to Koestler [19], in addition to metaphor, the well-known examples of mechanisms that can be used in cross-domain knowledge transfer are analogy and bisociation. Before addressing bisociative computational creativity, we continue with the presentation of a formal definition of bisociation, as formulated by Dubitzky [11].

Definition 1 (Domain theory) A domain theory D_i defines a set of concepts (knowledge units) that are associated with a particular domain i.

Definition 2 (Knowledge base) A knowledge base K_i is defined as a subset of a domain theory D_i; that is, $K_i \subseteq D_i$.

D_i denotes a domain theory which represents the total knowledge within a domain. The union of all domains then represents the universe of discourse: $\cup_i D_i = U$. Many domain theories overlap: $\exists i, j : D_i \cap D_j \neq \emptyset$. Let U denote the universe of discourse, which consists of all concepts. Let $c \in U$ denote a concept in U. Within U, a problem, idea, situation, or event π is associated with concepts $X \subset U$. Typically, a subset $P \subset X$ is used to reason about π.

Let R denote a reference system or intelligent agent which possesses exactly one knowledge base (empty or non-empty) per domain theory D_i. $K_i^R \in D_i$ denotes the knowledge base with respect to R and D_i.

$K^R = \cup_i K_i^R$ denotes the entire set of K incorporated in the reference system of R. K^R represents the total knowledge that R has in all the domains. For example, R may have non-empty knowledge bases for chess, but an empty one for geometry.

Definition 3 (Association) Let π denote a concrete problem, situation of event and let $X \subset U$ denote the concepts associated with π. Furthermore, let K_i^R denote an agent-specific knowledge base. Association occurs when elements of X are active or perceived in K_i^R at time t only.

For example, at time t, the concepts $A = \{c_1, c_2, c_3\}$ may be active in K_i^R only. In this case, we say that the concepts in A are associated.

Definition 4 (Habitually incompatible knowledge bases) Two agent-specific knowledge bases K_i^R and K_j^R $(i \neq j)$ are habitually incompatible if, at a given point in time t, there is no concept $c : c \in K_i^R \wedge c \in K_j^R$ that is active or perceived simultaneously in K_i^R and K_j^R.

Definition 5 (Bisociation) Let π denote a concrete problem, situation or event, and let $X \subset U$ denote the concepts associated with π. Furthermore, let K_i^R and K_j^R be such that $i \neq j$. Bisociation occurs when elements of X are active or perceived simultaneously in both K_i^R and K_j^R at a given point in time i.

For example, at time t, the concepts $B = \{c_1, c_2, c_3\}$ may be active or perceived simultaneously in K_i^R and K_j^R. In this case, the concepts in B are bisociated.

Bisociation cannot be equated with creativity in general. It is instead a special case of combinatorial creativity, which refers to novel combinations of familiar ideas: the creative aspect here is in the discovery of previously non-existing connections between domains, especially if each of the domains, or the elements repurposed from each, are very familiar. As put by Koestler [19], “the more familiar the parts, the more striking the new whole”. This is so because creation is never really a de novo nor random activity; it requires meaningful combination of elements.

Starting from Kostler’s [19] concept of bisociation, concrete bisociative patterns that are searched for in bisociative knowledge discovery include: bridging concepts, bridging graphs, and bridging by structural similarity [20]:

Bridging concepts	This is the most natural type of bisociation: a concept connecting two domains. In practice, different literatures from different domains are explored, and some terms connecting the two are found. This is the kind of pattern originally explored by Swanson. These connecting terms allow us to corroborate hypotheses linking the two domains. Bridging concept in the intersection of two domains A and C is illustrated in Fig. 9.
Bridging graphs	More complex bisociations are modeled by bridging graphs, in a network representation. This is similar to bridging concepts, but in this case, what connects two different domains is a subset of related concepts.
Bridging by structural similarity	This is the most complex kind of bisociation, whereby, again in a network representation, subsets of concepts in each domain share structural similarities, illustrated in Fig. 10.

Bisociations based on structural similarity are represented by relations and/or subgraphs of two different, structurally similar domains [20], as illustrated in Fig. 10. This type of bisociation is according to [20] the most abstract pattern with the potential for new cross-domain discoveries, which, e.g., vertex similarity methods can identify.

A special case of bridging by structural similarity is the concept of bridging by relational bisociation, as illustrated in Fig. 11, which will be explained and used in the novel methodology proposed in "Novel embeddings-based bisociative LBD methodology".

Word Embedding Potential for Creative Knowledge Discovery

Note that in this research, we neither use the TF-IDF representation of documents nor do we use document embeddings; instead, we focus on word embeddings. Word embeddings are vector representations of words: each word is assigned a vector of several hundred dimensions. These are usually obtained via training algorithms such as word2vec [25], GloVe [28], or FastText [4], which characterize the word based on the lexical context in which it appears. These representations improve performance in a wide range of automated text processing tasks, partly because they capture a degree of semantics. They can also capture regularities beyond simple relatedness, such as analogies [27]. A well-known example, illustrating this notion, is that word embeddings may explicitly find relations between words, as well as discover analogies between word pairs, such as that, e.g., the relation between Madrid and Spain is very similar to that between Paris and France in the embedded vector space (see Fig. 12).

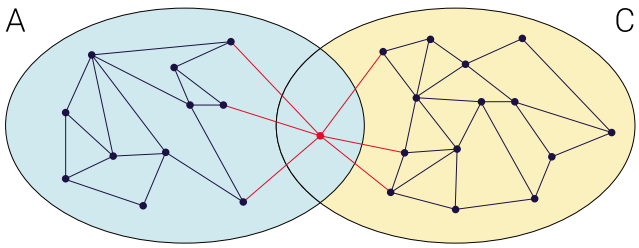


Fig. 9 Bridging concept in the intersection of two literature domains *A* and *C*

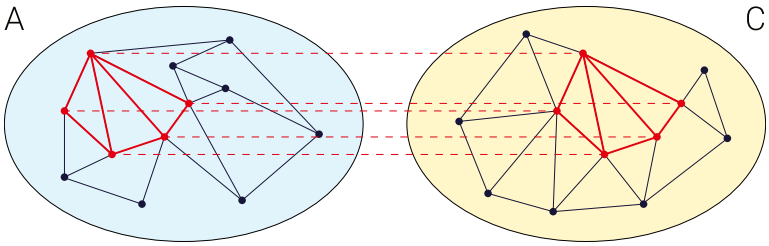


Fig. 10 Bridging by structural similarity of graphs [20]

Note that the analogies can be discovered within a single domain, as illustrated in Fig. 12. On the other hand, research in cross-lingual embeddings [8] has demonstrated the ability of aligning embeddings spaces across languages, which can be used as a basis for finding analogies across corpora in different languages [42], as investigated in the current EMBEDDIA EU project.[6]

In this paper, we propose a novel methodology, based on the idea of translating the cross-lingual setting to a cross-domain setting: instead of considering two different languages, we consider two separated domains *A* and *C*, use contemporary alignment methods [8] to align related concepts in the two domains, and finally perform analogy detection across the two domains [42]. In this way, we find bisociations by implementing the idea of bridging by relational bisociation.

Novel Embedding-Based Bisociative LBD Methodology

Most important for this paper is the property of word embeddings that they can capture regularities beyond simple relatedness, such as analogies [27], illustrated in Fig. 12. In the particular closed literature-based discovery setting of interest to this research, we implement the concept of bridging by relational bisociation.

Bridging by relational bisociation

We propose a particular setting of bridging by relational bisociation, illustrated in Fig. 11, where we are interested whether given a specific relation between two concepts $a1$ and $a2$

[6] www.embeddia.eu, see details in Acknowledgements.

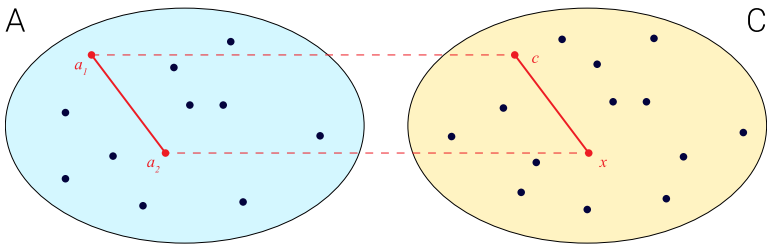


Fig. 11 Bridging by relational bisociation, the concept newly introduced in this paper

in first domain A, one can bisociatively discover an analogous relation between concepts x and c in second domain C, where c is a given concept and x is a new concept that we are trying to find. More formally, this can be written in the form of an analogy (i.e., bisociation) between two separate domains A and C as follows:

$$\mathtt{a1\,rel\,a2} :: \mathtt{x\,rel\,c}.$$

In the embeddings space, this analogy translates to the following equation between embeddings:

$$\mathtt{x} = \mathtt{emb(a1)} + \mathtt{emb(a2)} - \mathtt{emb(c)}.$$

Finally, once x is calculated, we need to find a set of concepts from the second domain C that have an embeddings representation most similar to x according to some predefined distance measure (e.g., the cosine similarity).

Methodology of bridging by relational bisociation

Proposed embedding-based bisociative LBD methodology for creative discovery of bisociated relationships between two domains A and C consists of the following steps:

1. Select two domains A and C, i.e., two document corpora such as circadian rhythm and plant defense, respectively.
2. Train separate word embeddings models for A and C to get $emb(A)$ and $emb(C)$.
3. Perform alignment of $emb(A)$ and $emb(C)$ embeddings vector spaces.
4. Determine the relationships of interest in a given domain A between concepts $a1$ and $a2$ defined by the biology expert.
5. Perform the embeddings-based relational LBD with a known seed concept c in C by leveraging the ability of the embeddings representations to model analogy relations.
6. Evaluate a list of best-ranked relational bisociations.

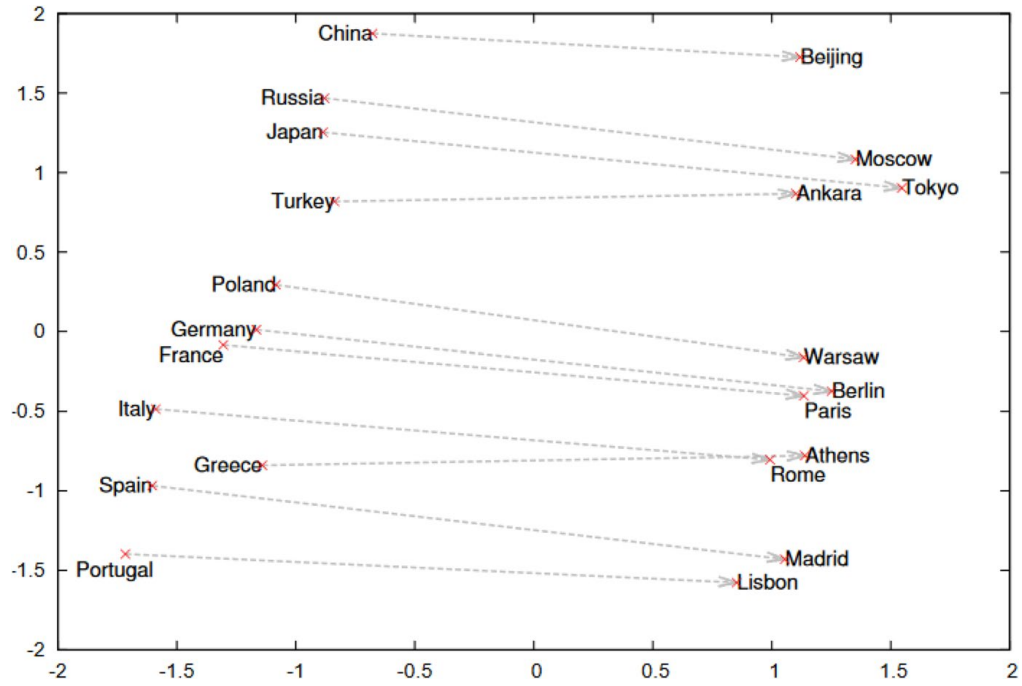


Fig. 12 Two-dimensional projection of embeddings illustrating capital–country relations. Picture taken from Mikolov [26]

Embeddings-Based Relational LBD Experiment Conducted on the Circadian Rhythm and Plant Defense Domains

In this section, we report in detail on the experiments conducted on the circadian rhythm and plant defense domains. Our main goal was to identify potentially interesting new daily regulated mechanisms that are responsible for plant defence. Circadian rhythm in plants causes that some of their genes are expressed differently during the course of the day. Consequently, plants respond differently to disease-causing infection if they are infected at different times of the day (e.g., morning, noon, and evening). Therefore, one of the goals of our study was to identify new gene sets that are differently expressed in different parts of the day and are important for the defense of plants against the pathogen.

After obtaining 10,494 documents from PubMed containing article titles and abstracts (4346 from plant defence and 6148 from circadian rhythm), we replaced gene names with synonyms gathered in previous research projects (22,265 gene names mapped into 7863 synonyms). In addition, we pre-processed the documents to keep only gene-related terms (included in synonym list and from the gene dictionary containing additional 6083 gene names), which resulted in a substantial reduction of the input document corpus, which we called the genesOnly dataset. The experiments that were conducted following the methodology proposed in "Novel embeddings-based bisociative LBD methodology" served as a proof of concept to show that the new proposed embeddings-based methodology can be used for LBD.

On each of the two selected domains, circadian rhythm and plant defense, we trained a separate FastText embedding model [4]. FastText embeddings were chosen due to their ability to leverage both semantic and morphological information by representing each word as an average of its character *n* grams. This is useful in a setting with a relatively small domain corpora containing less semantic information, since morphological similarity in many cases translates to semantic relatedness. We used a skip-gram model with an embedding dimension of 100.

The resulting embedding models trained for each domain were in the next step aligned into a common vector space. We opted for a supervised alignment approach, which relies on a training dictionary of identical words from both domains that are used as anchor points to learn a mapping from the source to the target space with a Procrustes alignment [8]. Train and test dictionaries were constructed by taking 5000 most frequent words from both domains (i.e., words that appear in both domain and have the largest sum of frequencies) and then split randomly into a train dictionary containing two-thirds of the words (3333) and a test dictionary containing one-third of the words (1667).

The success of the alignment was measured on the test dictionary in terms of precison@k, where precison@1 represents a share of model's correct alignments (exact matches) in a set of all alignments, and precison@5 represents a share of model's alignments in a set of all alignments, where the correct match for the word is found in the set of 5 most probable alignments predicted by the model. In the conducted experiment, we report the precison@1 of 0.4 and precison@5 of 0.55.

Next, we asked a biology expert to identify a list of genes related to the circadian rhythm domain. The following list was produced:

1. CCA1 = CIRCADIAN CLOCK ASSOCIATED1
2. LHY = LATE ELONGATED HYPOCOTYL
3. TOC1 = TIMING OF CAB EXPRESSION 1
4. PRR1 = PSEUDO-RESPONSE REGULATOR 1
5. GI = GIGANTEA
6. LNK1 = NIGHT LIGHT-INDUCIBLE AND CLOCK-REGULATED 1
7. PRR5 = PSEUDO-RESPONSE REGULATOR 5
8. ELF4 = EARLY FLOWERING 4
9. PRR9 = PSEUDO-RESPONSE REGULATOR 9
10. PRR7 = PSEUDO-RESPONSE REGULATOR 7
11. PCL1 = PHYTOCLOCK 1
12. ELF3 = EARLY FLOWERING 3

In addition, we also took more general key concepts from the circadian rhythm domain:

13. NEGATIVE FEEDBACK LOOP
14. OSCILLATOR
15. CLOCK.

According to the methodology explained in "Novel embeddings-based bisociative LBD methodology", we tried to identify a list of genes related to the concept of plant defense in the similar way the genes from the above list are related to the concept of circadian rhythm. First, we calculated embedding x according to the following equation:

$$\mathtt{x} = \mathtt{emb(a1)} + \mathtt{emb(a2)} - \mathtt{emb(c)},$$

where `a1` is a concept circadian rhythm, `a2` is a gene from the above list, and `c` is a concept plant defense.

Finally, once `x` was calculated for each of the genes from the above list, we searched for a set of concepts from the plant defense domain that have an embeddings representation most similar to `x` according to the cosine similarity. To limit the results only to genes or gene-related concepts, the concepts from the second domain were considered only if they appeared in the reduced genesOnly dataset. Ten genes or gene-related concepts with the representation most similar to each of the calculated `x`s were identified and given to the biology expert for the evaluation.

Results

The biology domain expert evaluated the selected set of output terms for all given analogy inputs. More specifically, for the analogies—a2 is as important to a1 (circadian rhythm) as x is to c (plant defense)—for each input relation, the resulting list of 10 candidates most similar to x (according to the cosine similarity between the candidate's embedding and x) were evaluated by the expert, who was given instructions to manually classify the relatedness between a candidate and the plant defense domain into the following four categories: NO, NOT AT ALL; NOT REALLY; MAYBE; YES. While YES is the category serving as a proof that the methodology works, MAYBE is the category containing very interesting terms from the knowledge discovery point of view, as, here, the experts might potentially search for novel knowledge.

First, we calculated the average precision at 10 (p@10) for each output list of 10 candidates, as well as a microaveraged precision for the entire dataset (see Table 2). We can observe that the method performed very well. In 40% of the cases, the expert found in the scientific literature that the discovered relation between the plant defense concept and the proposed term x is meaningful. We can see that precision varies for different input relations, but the method was able to find at least one correct relation in the plant defense domain for each circadian rhythm input relation. For input relations between the concept circadian rhythm and genes ELF4, PRR9 and PRR7, six out of ten term candidates in the resulting candidate lists are related to the plant defense domain. On the other hand, the lowest results are for the input concept negative feedback loop, where only for one out of ten output terms, the expert found that the output term was relevant for the domain. A reason for this could be that the input term is one of the few terms, which is not a gene but rather a gene-related concept (text), and that it is a multi-word expression, for which the average embedding was first calculated (by averaging embeddings for each word in

the term) to obtain the term embedding, and, therefore, the results might be less precise.[7]

For the category MAYBE, which is the most interesting category for the new knowledge discovery and for which the outputs might possibly be investigated in detail in the future research by the domain experts, we can note that for all input relations but one, at least one out of 10 outputs was considered potentially interesting. In a knowledge discovery setting, where each discovery if resulting in new domain knowledge would have big impact, this was considered as a promising result.

As explained above, the biology expert evaluated 10 candidates for each input relation. These relations were ranked according to the cosine similarity between x and the candidate, with rank 1 representing the candidate closest to x, i.e., with the largest cosine similarity to x. Table 3 presents results for candidates with different ranks. Note that here we measured precision at 15, i.e., how many out of 15 predicted terms with a specific rank had been evaluated as related to the plant defense domain (P@15 yes) or as maybe being related to the plant defense domain (P@15 maybe). Interestingly, the correlation between precision at 15 and rank was not strong and better ranked candidates were not necessarily more correlated to the plant defense domain according to the evaluation. For example, the best evaluated candidates had rank 5, where P@15 yes was 0.6 and P@15 maybe was 0.133.

Next, we removed duplicate outputs, merged all the output terms from all the inputs, and calculated the class distribution for this list (see Table 4). The rationale for this procedure is that since, in our case, the a1 and c were always the same (equivalent to domain names circadian rhythm and plant defence) and as the different a2 all modeled the same relation—a2 is as important to a1 (circadian rhythm) as x is important to c (plant defense)—we could treat also all results as a common list of relevant terms (genes). As the results indicate, about 37% of output terms were evaluated as relevant to the plant defense domain. Also, together with the category MAYBE, which indicates that the output is potentially relevant (but requires further research), this percentage of relevant terms increased to nearly 55%.

From 40 examples in the categories YES and MAYBE, 32 were gene names and 3 were proteins, while the rest referred to a disease or partial names of proteins, genes, etc. Below, we list ten terms and their full names that were classified in category YES and appeared in results of at least 3 input terms:

1. DMR1 = DOWNY MILDEW RESISTANT 1
2. CPR30 = CONSTITUTIVE EXPRESSER OF PR GENES 1
3. EIF4G = EUKARYOTIC TRANSLATION INITIATION FACTOR 4 G
4. SLAC1 = SLOW ANION CHANNEL-ASSOCIATED 1

[7] There are several possible multi-word expression aggregation approaches, such as summation of component word vectors, averaging of component word vectors, creating multi-word term vectors, etc. As comparing different techniques is beyond the scope of this study, we decided for the simple averaging technique, as the previous research on this topic conducted on the medical domain [14] found no statistically significant difference between any multi-word expression aggregation method.

Table 2 Evaluation for 15 input relations

Source gene/term	No, not at all	Not really	Maybe	Yes	P@10 Maybe	P@10 Yes
CCA1	1	3	1	5	0.1	0.5
LHY	0	5	1	4	0.1	0.4
TOC1	0	1	3	6	0.3	0.6
PRR1	0	5	2	3	0.2	0.3
GI	1	5	2	2	0.2	0.2
LNK1	1	4	1	4	0.1	0.4
PRR5	0	3	3	4	0.3	0.4
ELF4	0	2	2	6	0.2	0.6
PRR9	0	4	0	6	0.0	0.6
PRR7	0	3	1	6	0.1	0.6
PCL1	2	6	0	2	0.0	0.2
ELF3	1	4	2	3	0.2	0.3
NEGATIVE FEEDBACK LOOP	7	0	2	1	0.2	0.1
OSCILLATOR	4	2	1	3	0.1	0.3
CLOCK	1	1	3	5	0.3	0.5
All	18	48	24	60	0.16	0.4

5. RFC3 = REPLICATION FACTOR C SUBUNIT 3
6. RTM1 = RESTRICTED TEV MOVEMENT 1
7. SNI1 = SUPPRESSOR OF NPR1-1
8. GRF6 = GROWTH-REGULATING FACTOR 6
9. NAC083 = NAC DOMAIN CONTAINING PROTEIN 83
10. XAP5 = XAP5 CIRCADIAN TIMEKEEPER

Summary and Lesson Learned from These Experiments

Given the proof-of-concept evaluation of these results, the proposed methodology demonstrates its relevance for knowledge discovery research. One of the most interesting findings observed from the conducted experiments was the presence of some resistance and susceptibility genes among the candidates proposed by the method; these genes are known to play an important role in the plant defense process. Moreover, the best ranked candidate obtained for the *c* term inputs CCA1 and LHY (two central genes of the circadian clock rhythm) was DMR1 (a susceptibility gene, mutation of this gene results in a higher resistance), that is a hot topic of a plant resistance research lately. In future work, the genes identified in results will be closely inspected by domain experts. In conclusion, let us summarize this section by the lesson learned from these experiments.

Lesson Learned 6: Term filtering and synonyms matter In the experiments using plant defence-circadian

Table 3 Evaluation according to rank

Rank	No, not at all	Not really	Maybe	Yes	P@15 Maybe	P@15 Yes
1.	2	4	1	8	0.067	0.533
2.	2	6	1	6	0.067	0.400
3.	1	7	1	6	0.067	0.400
4.	3	7	2	3	0.133	0.200
5.	1	3	2	9	0.133	0.600
6.	4	1	3	7	0.200	0.467
7.	1	4	4	6	0.267	0.400
8.	2	3	5	5	0.333	0.333
9.	1	6	3	5	0.200	0.333
10.	1	7	2	5	0.133	0.333
All	18	48	24	60	0.160	0.400

Table 4 Evaluation on all output terms (duplicates removed)

Label	Count	Perc. (%)
NO, NOT AT ALL	14	19.18
NOT REALLY	19	26.03
MAYBE	13	17.81
YES	27	36.99
Total	73	100

rhythm domain pair, the goal was to identify potentially interesting new daily regulated mechanisms that are responsible for plant defence. After obtaining 5412 documents from PubMed containing complete articles (2483 from plant defence and 2929 from circadian rhythm), 0.5% documents shorter than 20 characters (mostly empty contents) and longer than 97,500 characters (containing many different articles in proceedings) were removed. Then, 12 duplicates that were present in both domains (as in Lesson Learned 2) were eliminated. The crucial, although simple and straightforward, step in this experiment was the replacement of gene names with synonyms gathered in the previous research projects (22,265 gene names mapped into 7863 synonyms). In addition, the documents were optionally pre-processed to keep only gene-related terms (included in synonym list and from the gene dictionary containing

additional 6083 gene names), which resulted in a substantial reduction of the input file size (from 200 to 28 MB).

Conclusions and Further Work

This paper addresses the field of scientific computational creativity, in particular bisociative literature-based discovery. The paper mostly focused on finding outlier documents as means for finding unexpected links crossing different contexts. Selected approaches to bridging term detection through outlier document exploration are briefly outlined, together with the lessons learned from recent applications in medical and biological literature-based knowledge discovery. Finally, the paper addresses new prospects in bisociative literature-based discovery, proposing a novel methodology exploiting the use of advanced embedding technology for bisociative cross-domain literature mining.

Our future work, aimed at improving the effectiveness of bridging term detection in cross-domain literature mining, will be performed in several directions, based on our current research: using ontologies for term enrichment in cross-domain document exploration, and using network analysis for cross-domain heterogeneous information network exploration.

- The use of background knowledge remains largely unexploited in text classification and clustering. Word taxonomies can easily be exploited as means for constructing new semantic features, which can be used in the text representation learning to improve the performance and robustness of the learned models. Consequently, our novel tax2vec algorithm [34] could be used for constructing taxonomy-based features to improve the results of document clustering and classification.
- Given that documents can be easily transformed into graphs (e.g., graphs constructed from subject–verb–object triplets), network analysis approaches can prove to be fruitful for bridging term detection (e.g., community detection and finding bridging nodes in graphs between subgraphs representing the detected communities).
- We will also introduce additional user-interface options for data visualization and exploration, as well as advance our bridging term ranking methodology [17] by adding new heuristics, which will take into account also the semantic aspects of the data.
- Most importantly, we will further explore embeddings-based LBD in the closed LBD settings, aiming to improve and further explore the methodology proposed in "Towards creative embeddings-based bisociative LBD". Especially, we plan to focus on bisociative discovery without known concept c, as well as on enabling multi-word expressions as output.
- We will experiment with new application topics. It will be especially insightful to address problems in need of discovering novel bisociations between two

different domains. Also, it could be useful to investigate two entirely unrelated domains to provide a baseline.

Author Contributions All authors contributed to the study conception and design. The first draft of the manuscript was written by Nada Lavrač, all authors contributed to manuscript writing, data collection, and analysis were performed by Matej Martinc, Senja Pollak, Maruša Pompe Novak, and Bojan Cestnik. All authors approved the final manuscript.

Funding This work was supported by the Slovenian Research Agency (ARRS) grants Knowledge technologies P2-0103, and Terminology and knowledge frames across languages (J6-9372), and the European Union's Horizon 2020 research and innovation programme under grant agreement No 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media).

Availability of data and material (data transparency) Available upon request from the second author Matej Martinc.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

Code availability (software application or custom code) Open source.

References

1. Abgaz, Y., O'Donoghue, D., Hurley, D., Smorodinnikov, D.: Evaluation of analogical inferences formed from automatically generated representations of scientific publications. In: 24th Irish Conference on Artificial Intelligence and Cognitive Science (2016)
2. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I., et al.: Fast discovery of association rules. Adv. Knowl. Discov. Data Min. **12**(1), 307–328 (1996)
3. Berthold, M. (ed.): Bisociative Knowledge Discovery. Springer, Berlin (2012)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Trans. Assoc. Comput. Linguist. **5**, 135–146 (2017)
5. Brodley, C.E., Friedl, M.A.: Identifying mislabeled training data. J. Artif. Intell. Res. **11**, 131–167 (1999)
6. Bruza, P., Weeber, M.: Literature-based Discovery. Springer Science & Business Media, Berlin (2008)

7. Cestnik, B., Fabbretti, E., Gubiani, D., Urbančič, T., Lavrač, N.: Reducing the search space in literature-based discovery by exploring outlier documents: a case study in finding links between gut microbiome and Alzheimer's disease. Genom. Comput. Biol. **3**(3), e58 (2017)
8. Conneau, A., Lample, G., Ranzato, M., Denoyer, L., Jégou, H.: Word translation without parallel data. CoRR abs/1710.04087 (2017)
9. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. (2018). arXiv preprint arXiv:1810.04805
10. Dong, F., O'Donoghue, D., Ersotelos, N., Wu, S., Saggion, H., Ronzano, F., Corcho, Ó., Hurley, D., Abgaz, Y.M., Zhang, J., Chaudhry, E., Yang, X., Wei, H., Deng, Z., Mahdian, B., Careil, J.M.: Dr. inventor, promoting scientific creativity by utilising web-based research objects. Impact **2**, 40–44 (2017)
11. Dubitzky, W., Kötter, T., Schmidt, O., Berthold, M.R.: Towards creative information exploration based on Koestler's concept of bisociation. In: Berthold, M.R. (ed.) Bisociative Knowledge Discovery: An Introduction to Concept, Algorithms, Tools, and Applications, pp. 11–32. Springer, Berlin (2012)
12. Fortuna, B., Grobelnik, M., Mladenić, D.: Semi-automatic data-driven ontology construction system. In: Proceedings of the 9th International Multi-conference Information Society, pp. 223–226 (2006)
13. Gopalakrishnan, V., Jha, K., Jin, W., Zhang, A.: A survey on literature based discovery approaches in biomedical domain. J. Biomed. Inform. **93**, 103141 (2019)
14. Henry, S., Cuffy, C., McInnes, B.T.: Vector representations of multi-word terms for semantic relatedness. J. Biomed. Inform. **77**, 111–119 (2018)
15. Holzinger, A., Yildirim, P., Geier, M., Simonic, K.M.: Quality-based knowledge discovery from medical text on the web. In: Quality Issues in the Management of Web Information, pp. 11–13. Springer (2013)
16. Hristovski, D., Peterlin, B., Mitchell, J.A., Humphrey, S.M.: Using literature-based discovery to identify disease candidate genes. Int. J. Med. Inform. **74**(2), 289–298 (2005)
17. Juršič, M., Cestnik, B., Urbančič, T., Lavrač, N.: Cross-domain literature mining: Finding bridging concepts with CrossBee. In: Proceedings of the 3rd International Conference on Computational Creativity, pp. 33–40 (2012)
18. Kastrin, A., Rindflesch, T.C., Hristovski, D.: Link prediction on the semantic MEDLINE network. In: Proceedings of the International Conference on Discovery Science, pp. 135–143. Springer (2014)
19. Koestler, A.: The Act of Creation. Hutchinson, Paris (1964)
20. Kötter, T., Berthold, M.: From information networks to bisociative information networks. In: Bisociative Knowledge Discovery, pp. 33–50. Springer (2012)
21. Lavrač, N., Juršič, M., Sluban, B., Perovšek, M., Pollak, S., Urbančič, T., Cestnik, B.: Bisociative knowledge discovery for cross-domain literature mining. In: Veale, T., Cardoso, F.A. (eds.) Computational Creativity: The Philosophy and Engineering of Autonomously Creative Systems, pp. 121–139. Springer, Berlin (2019)
22. Lavrač, N., Martinc, M., Pollak, S., Cestnik, B.: Bisociative literature-based discovery: Lessons learned and new prospects. In: Proceedings of International Conference on Computational Creativity (2020) (**In press**)
23. Lindsay, R.K., Gordon, M.D.: Literature-based discovery by lexical statistics. J. Am. Soc. Inf. Sci. Technol. **1**, 574–587 (1999)
24. Martinc, M., Škrlj, B., Pirkmajer, S., Lavrač, N., Cestnik, B., Marzidovšek, M., Pollak, S.: Covid-19 therapy target discovery with context-aware literature mining. In: Proceedings of International Conference on Discovery Science. Springer (2020) (**In press**)
25. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of ICLR CoRR. abs/1301.3781 (2013)
26. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Welling, M., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems, vol. 26, pp. 3111–3119. Curran Associates Inc., New York (2013)
27. Mikolov, T., Yih, W.T., Zweig, G.: Linguistic regularities in continuous space word representations. In: Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 746–751. Association for Computational Linguistics (2013)

28. Pennington, J., Socher, R., Manning, C.: Glove: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543. Association for Computational Linguistics, Doha, Qatar (2014)
29. Perovšek, M., Kranjc, J., Erjavec, T., Cestnik, B., Lavrač, N.: TextFlows: a visual programming platform for text mining and natural language processing. Sci. Comput. Program. **121**, 128–152 (2016)
30. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. (2018). arXiv preprint arXiv:1802.05365
31. Petrič, I., Cestnik, B., Lavrač, N., Urbančič, T.: Outlier detection in cross-context link discovery for creative literature mining. Comput. J. **55**(1), 47–61 (2012)
32. Petrič, I., Urbančič, T., Cestnik, B., Macedoni-Lukšič, M.: Literature mining method rajolink for uncovering relations between biomedical concepts. J. Biomed. Inform. **42**(2), 219–227 (2009)
33. Sebastian, Y., Siew, E.G., Orimaye, S.O.: Emerging approaches in literature-based discovery: techniques and performance review. Knowl. Eng. Rev. **32**, e12 (2017)
34. Škrlj, B., Martinc, M., Kralj, J., Lavrač, N., Pollak, S.: tax2vec: constructing interpretable features from taxonomies for short text classification. Comput. Speech Lang. **65**, 101104 (2021)
35. Sluban, B., Gamberger, D., Lavrač, N.: Ensemble-based noise detection: Noise ranking and visual performance evaluation. Data Min. Knowl. Discov. **28**, 1–39 (2013)
36. Sluban, B., Juršič, M., Cestnik, B., Lavrač, N.: Exploring the power of outliers for cross-domain literature mining. In: Bisociative Knowledge Discovery, pp. 325–337. Springer (2012)
37. Smalheiser, N., Swanson, D.R.: Using ARROWSMITH: a computer-assisted approach to formulating and assessing scientific hypotheses. Comput. Methods Programs Biomed. **57**(3), 149–154 (1998)
38. Srinivasan, P.: Text mining: generating hypotheses from MEDLINE. J. Am. Soc. Inf. Sci. Technol. **55**(5), 396–413 (2004)
39. Swanson, D.R.: Migraine and magnesium: eleven neglected connections. Perspect. Biol. Med. **78**(1), 526–557 (1988)
40. Swanson, D.R.: Medical literature as a potential source of new knowledge. Bull. Med. Lib. Assoc. **78**(1), 29 (1990)
41. Swanson, D.R., Smalheiser, N.R., Torvik, V.I.: Ranking indirect connections in literature-based discovery: the role of medical subject headings (MeSH). J. Am. Soc. Inf. Sci. Technol. **57**(11), 1427–1439 (2006)
42. Ulčar, M., Robnik-Šikonja, M.: Multilingual culture-independent word analogy datasets. In: Proceedings of LREC (2020) (**In press**)
43. Weeber, M., Klein, H., de Jong-van den Berg, L., Vos, R., et al.: Using concepts in literature-based discovery: simulating Swanson's Raynaud-fish oil and migraine-magnesium discoveries. J. Am. Soc. Inf. Sci. Technol. **52**(7), 548–557 (2001)
44. Yetisgen-Yildiz, M., Pratt, W.: Using statistical and knowledge-based approaches for literature-based discovery. J. Biomed. Inform. **39**(6), 600–611 (2006)

Appendix C: COVID-19 therapy target discovery with context-aware literature mining

COVID-19 therapy target discovery with context-aware literature mining

Matej Martinc[1,2], Blaž Škrlj[1,2], Sergej Pirkmajer[3], Nada Lavrač[1,2,4], Bojan Cestnik[5,1], Martin Marzidovšek[1,2], and Senja Pollak[2]

[1] Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
[2] Jožef Stefan Institute, Ljubljana, Slovenia
[3] Institute of Pathophysiology, Faculty of Medicine, University of Ljubljana, Ljubljana, Slovenia
[4] University of Nova Gorica, Vipava, Slovenia
[5] Temida d.o.o, Ljubljana, Slovenia

The final reviewed publication was published in Proceedings of the 23rd International Conference on Discovery Science (DS 2020), Thessaloniki, Greece, October 19–21, 2020 and is available online at https://doi.org/10.1007/978-3-030-61527-7_8.

Abstract. The abundance of literature related to the widespread COVID-19 pandemic is beyond manual inspection of a single expert. Development of systems, capable of automatically processing tens of thousands of scientific publications with the aim to enrich existing empirical evidence with literature-based associations is challenging and relevant. We propose a system for contextualization of empirical expression data by approximating relations between entities, for which representations were learned from one of the largest COVID-19-related literature corpora. In order to exploit a larger scientific context by transfer learning, we propose a novel embedding generation technique that leverages SciBERT language model pretrained on a large multi-domain corpus of scientific publications and fine-tuned for domain adaptation on the CORD-19 dataset. The conducted manual evaluation by the medical expert and the quantitative evaluation based on therapy targets identified in the related work suggest that the proposed method can be successfully employed for COVID-19 therapy target discovery and that it outperforms the baseline FastText method by a large margin.

Keywords: Knowledge discovery · Literature mining · Representation learning · Contextual embeddings · COVID-19.

1 Introduction

Scientific knowledge for a specific domain is in most cases given in an unstructured form, as a set of scientific papers covering a variety of findings, experiments

and methodologies related to a specific scientific field or problem. The current speed and quantity of scientific research production makes manual inspection of the literature from a specific field virtually impossible. The recent trend of interdisciplinary research complicates things even more, as it would require from a researcher to understand all the aspects, from which a specific research problem can be covered in order to "connect all the dots" and advance the field by the discovery of the so-called latent scientific knowledge.

To solve this problem, several automated strategies for uncovering this knowledge have been proposed. Somewhat older studies proposed literature-based discovery (LBD) [8] focusing especially on cross-domain literature mining, which aims at finding interesting bridging terms (b-terms) or bridging links revealing the potentially new connections between separate domain corpora of interest. On the other hand, more recent approaches to latent knowledge discovery from the scientific literature employ word embeddings [26]. For example, a study by [34] showed that latent knowledge regarding future discoveries is to a large extent embedded in past publications by retrieving information from the scientific literature with the usage of Word2Vec embeddings [26].

The latest development in the natural language processing (NLP) is a new type of embeddings called contextual embeddings. ELMo (Embeddings from Language Models) [29] and BERT (Bidirectional Encoder Representations from Transformers) [12] are the most prominent representatives of this type of contextual embeddings, and have been also adapted to scientific literature [3]. The main difference between these novel contextual embeddings and older "static" embeddings is that in these embeddings a different vector is generated for each context a word appears in, i.e., for each specific word usage in the corpus. These new contextual embeddings solve the problems with word polysemy and other changes in word meaning given different context. On the other hand, it is not entirely clear how to generate a meaningful general word representation from the word usage embeddings. This means that the usage of contextual embeddings for LBD is not entirely straight forward, since they can not be used in the same way as the traditional static embeddings, and have at least to our knowledge not been used for the task at hand.

In this work, we explore how contextual embeddings can be leveraged for the task of discovering latent scientific knowledge in the very topical scientific literature about the COVID-19 disease. More specifically, we are interested in the discovery of new COVID-19 therapy targets from the targets discovered in the past research. The novelty of this work is two-fold:

- The paper contributes a new methodology of generating general word representations from contextual embeddings, proposes an entire workflow for acquisition of novel COVID-19 therapy targets and shows that our method of using contextual embeddings for LBD outperforms the baseline method of using static embeddings by a large margin.
- Medically, the paper contributes to identifying new potential COVID-19 therapy targets, motivated by a recent proof-of-concept study that used a

state-of-the-art omics approach to identify new possible targets for existing drugs, such as ribavirin [5].

2 COVID-19 medical background and recent therapy targets

In late 2019 a novel coronavirus disease (COVID-19), caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2), emerged in China [38,39]. COVID-19 quickly spread and was declared a pandemic by the World Health Organization.

While new targeted therapies and vaccines against SARS-CoV-2 virus are being actively developed, their potential use in the clinics is not imminent. Therefore, until effective pharmacological therapies and/or vaccines are available, medicine needs to resort to other approaches to treat patients with COVID-19 or prevent transmission of SARS-CoV-2. One approach is to identify which among the antiviral drugs that were developed to treat other viral diseases might be effective against SARS-CoV-2. A preliminary report suggests that remdesivir seems to be the most promising candidate among these drugs [2]. Another approach is to identify drugs that are used for other purposes but also exert antiviral effects. The most prominent example among these is hydroxychloroquine, which is used for chronic treatment of rheumatic diseases but also suppresses SARS-CoV-2 in vitro [22]. Identifying a known drug with well-characterized adverse effects would certainly save time and lives before more specific treatments are developed. However, repurposing of existing drugs is also a challenge as highlighted by a recent controversy with hydroxychloroquine [7,25] and new candidate drugs and/or therapeutic targets are needed.

3 Related work

The related work is divided into three Sections, namely related work on Literature-based discovery in Section 3.1, related work on text representation learning in Section 3.2 and selected overview of recent NLP research on COVID-19 in Section 3.3.

3.1 Literature-based discovery

Literature-based discovery (LBD) aims to generate new knowledge by combining what is already known in the literature. It has been used to (semi-automatically) identify new connections between genes, drugs and diseases, etc. [18]. Traditionally, LBD has been addressed as finding interesting bridging terms revealing the potentially new connections between separate domain corpora of interest [8]. Swanson [33] developed one of the early LBD approaches, the so-called ABC model, to detecting interesting b-terms to uncover the possible cross-domain relations among previously unrelated concepts.

On the other hand, a more recent state-of-the art tool LION LBD [31] enables researchers to navigate published information and supports hypothesis generation and testing. The system is built with a particular focus on the molecular biology of cancer. LBD has led to discovery of potential treatments in other domains, including multiple sclerosis [19], and has been applied successfully in drug development and repurpusing [11]. Recent LBD approaches benefit from word embeddings. One is the study by [34] already mentioned in Section 1 and the other is the work by [9], who proposed graph-based, neural network methods to perform open and closed LBD and demonstrated improved performance on existing tasks.

3.2 Text representation and embeddings

Recently, the embedding approach became a prevalent way to build representations for many different types of entities, e.g., texts, graphs, electronic health records, images, relations, recommendations, etc. Text embeddings use large corpora of documents to extract vector representations for words, sentences, and documents. The first neural word embeddings like Word2vec [26] produced one vector for each word, irrespective of its polysemy. These so-called static embeddings have been further developed and the most popular static embeddings currently in use besides Word2Vec are GloVe (Global vectors for word representation) [28] and FastText [4]. Recent developments like ELMo [29] and BERT [12] take a context of a sentence into account and produce different word vectors for different contexts of each word. Another novelty of these approaches is the employment of the transfer learning technique, which has recently become a well established procedure in the field of NLP. This procedure relies on a language model pretraining on very large unlabeled textual resources and after that transfer of the knowledge obtained by the language model onto a specific downstream task by further fine-tuning the model.

3.3 Text mining and NLP research related to COVID-19

With regard to biomedical research on COVID-19, time is a central factor as scientists try to design treatments and vaccines amid the pandemic caused by the SARS-CoV-2 virus, therefore leveraging LBD and its potential to reduce scientific discovery time could prove crucial.

Many search platforms emerged for retrieving COVID-19 related papers. For example, Neural Covidex[6] is based on neural ranking architecture and provides information access capabilities to the COVID-19 Open Research Dataset (CORD-19) (see Section 4.1). SciSight [17] in contrast to standard targeted search facilitates finding connections between biomedical concepts that are not obvious from reading individual papers. It displays a network of top related terms mined from the corpus, based on the co-appearance in the same sentence.

[6] https://covidex.ai/

Studies that can generate new knowledge about COVID-19 by applying embeddings are still scarce but do exist. For example, a recent study has projected Covid-related medical texts in a 3D human atlas space that helps to navigate the literature [14]. The objective was to learn semantically aware groundings of sentences with five different BERT models [12].

4 Background knowledge and resources

We describe the CORD-19 corpus (Section 4.1) and embeddings technology (Section 4.2) used in this study.

4.1 CORD19 database

The scientific literature considered in this work has been recently introduced as the CORD-19 corpus[7]. CORD-19 is a resource of over 135,000 scholarly articles, including over 68,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses. This freely available data set is provided to the global research community to apply recent advances in NLP and other AI techniques to generate new insights in support of the ongoing fight against this infectious disease.

We use the corpus version 12, published on May 1st 2020, from which we extract only full text scholarly articles converted into xml from a pdf format. This results in altogether 48,410 papers, which are summarized in Table 1.

Table 1. CORD-19 dataset statistics.

Origin	Number of papers	Number of tokens
Commercial use subset	9,918	46,206,453
Non-commercial use subset	2,584	10,732,608
PMC custom license subset	32,450	156,247,363
bioRxiv (not peers reviewed)	2,670	8,968,183
medRxiv subset (not peer reviewed)	788	3,285,558
All	48,410	225,440,165

4.2 Considered embeddings

We use FastText [4] embeddings as a baseline in this study. The main advantage of FastText embeddings is its word representation as a sum of n-grams, which allows the model to, in addition to leveraging semantic relations, also leverage morphological information.

One of the most oftenly used models for the generation of contextual embeddings is the BERT model [12] that was originally pretrained on the Google

[7] https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge

Books Corpus (800 million tokens) and Wikipedia (2,500 million tokens). This pretraining is however not entirely appropriate for the text mining tasks on the scientific literature due to specificities of the scientific language and vocabulary. For this reason, in this research we opted for SciBERT [3], a version of BERT pretrained on a large multi-domain corpus of scientific publications, a random sample of 1.14M papers from Semantic Scholar. SciBERT model has 12 encoder layers with the attention mechanism and a hidden layer size of 768.

5 Methodology

In this section, we present the methodology of the proposed approach by explaining how we obtain word representations, how we acquire therapy target candidates and how we evaluate the approach.

5.1 Word representations

First, we fine-tune SciBERT as a masked language model for domain adaptation on the lowercased CORD-19 dataset. Next, we generate word representations for each word in the vocabulary. Figure 1 visualizes the process described below. The documents from the corpus are split into sequences of byte-pair encoded tokens [20] of a maximum length of 256 tokens and fed into the fine-tuned SciBERT model. For each of these sequences of length n, we create a sequence embedding by summing the last four encoder output layers. The resulting sequence embedding of size n times *embeddings size* represents a concatenation of contextual embeddings for the n tokens in the input sequence. By chopping it into n pieces, we acquire a representation, i.e. a contextual token embedding, for each word used in the corpus. Note that these representations vary according to the context in which the token appears, meaning that the same word has a different representation in each specific context (sequence).

Finally, the resulting embeddings are aggregated on the token level (i.e. for every token in the corpus vocabulary, we create a list of all their contextual embeddings) and are averaged, in order to get one representation for each token in the vocabulary. We enforce a constraint that a list of contextual embeddings for a specific token should contain at least five elements, otherwise the specific token is discarded. This is done in order to remove tokens that do no appear in the corpus enough times for the model to learn a meaningful representation (e.g., mostly tokens that contain typos or very rare technical terms). Since the byte-pair input encoding scheme [20] employed by the SciBERT model does not necessarily generate tokens that correspond to words but rather generate tokens that correspond to parts of words, we also propose the following *on the fly* reconstruction mechanism that allows us to get word representations from byte pair tokens. If a word is split into more than one byte pair token, we take an embedding for each byte pair token constituting a word and build a word embedding by averaging these byte pair tokens. The resulting average is used as a context specific word representation.

The final result are static embeddings for each word in the vocabulary, capable of leveraging a broader semantic knowledge due to the SciBERT being pretrained on a large corpus of scientific articles. As a baseline, we also train a FastText skip-gram model with an embedding dimension of 100 (which is the default) on the lowercased CORD-19 dataset. Once again we enforce the constraint that a word should appear in the corpus at least five times.

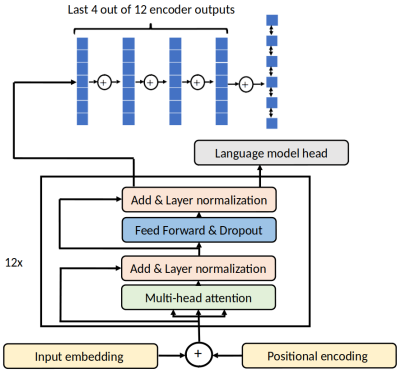


Fig. 1. Extraction of word usage embeddings from BERT. Note that only the last 4 out of 12 BERT encoder layers are used for the embedding generation. This was done in accordance with the previous studies that suggested that the last four layers carry the bulk of the semantic information obtained by the model [24].

5.2 Synonym resolution

Once embeddings are generated, we conduct synonym resolution with the help of a list of 19,302 gene names and their most common synonyms [30]. The embedddings belonging to the synonyms of the same gene are averaged in order to combine contextual information of different identifiers referring to the same gene and in order to avoid possible mismatches due to different naming.

5.3 Candidate acquisition

The main idea of our approach is to leverage semantic similarity in order to derive new scientific knowledge from an already existing one. For this to work, some initial seed concepts need to be acquired and used as a starting point. We explore two possibilities for this:

– **Seed concepts recommended by the expert**: The experts with a medical background were asked to recommend genes and/or proteins with a

known and confirmed link to COVID-19. The final consensus was to focus on angiotensin-converting enzyme 2 (ACE2) and transmembrane protease serine 2 (TMPRSS2). ACE2, a receptor for the spike S protein, is important because SARS-CoV-2 uses it to enter the host cell [16]. TMPRSS2 promotes SARS-CoV-2 entry into the cell by priming the spike S protein [16]. Blockage of binding of SARS-CoV-2 to ACE2 or inhibition of TMPRSS2 are therefore two possible approaches to treat COVID-19.

– **Seed concepts found in the literature**: Due to the abundance of recent research on COVID-19 it is also possible to find seed concepts in the related research. We opted for a study by [5] in which a set of COVID-19 therapy targets were identified. The considered list of altogether 2802 potential targets[8] is the result of a large-scale screening for active proteins, and offers a starting set of candidates obtained empirically. The list is ranked according to the increase or decrease of production of a specific protein at a specific time point. We explore what is the optimal number of seed candidates by exponentially enlarging the size of the seed candidate set. Sampling from the list is conducted according to the ranking of the protein candidates, i.e., we sample 2, 4, 8, 16, 32 and 64 best ranked seed candidates according to the increase in their production 24 hours after the infection (column Ratio 24h in Supplementary Table 1 in the study by [5]).

Once seed concepts are acquired, we calculate their embeddings and look for semantically similar concepts by finding the concepts that are the closest to seed concepts according to the cosine distance between the embeddings[9]. More specifically, we find a set of 2802 closest candidate concepts for each gene/protein in each seed candidate set, and the acquired candidates are ranked according to the cosine similarity. Finally, we calculate the average ranking for each candidate (i.e. by averaging ranks for each seed concept in the set) and therefore obtain NumOfCandidatesInSet $*$ 2802 closest candidates for each of the seed concept sets with possible duplicates originating from different seed concepts.

Since the initial experiments showed that many of the most similar concepts are in fact variations of the same base concept (e.g., the closest neighbours to ACE2 being ACE, ACE2M, ACE2S...) and since we are interested in maximizing the variety of the acquired candidates, we conduct an additional filtering according to the normalized Levenshtein distance defined as:

$$\text{NORMLD} = 1 - \frac{LD}{\max(\text{len}(w_1), \text{len}(w_2))},$$

where NORMLD stands for normalized Levenshtein distance, LD for Levenshtein distance, w_1 is either a seed concept or a concept already in the list of acquired

[8] Note that the original list contains 2715 targets (see Supplementary Table 1 in [5]). Some of them are however represented as a set of similar genes/proteins belonging to the same family. On the other hand, we treat each individual gene/protein as a separate target, which results in a set of 2802 targets.

[9] Note that these concepts obtained according to semantic similarity are not necessarily proteins/genes but rather any word in the embedding vocabulary.

neighbours and w_2 is the new candidate neighbour. Concepts for which normalized Levenshtein difference is bigger than 0.7 are discarded[10]. The filtering is conducted in order from the top of the list (neighbours with the best average rank) to the bottom.

At the end of the candidate acquisition process, we cut the ranked list of neighbours at 2802 target candidates for each of the distinct seed concept sets used in the evaluation.

5.4 Evaluation

The methods for discovering new therapy targets are evaluated in two evaluation settings, quantitative and qualitative.

Quantitative Evaluation We evaluate if therapy target candidates acquired in the previous step have been confirmed as targets in the study by [5], i.e. how many of them appear in the list of 2802 candidates they identified[11]. Note that in this setting we only evaluate the proposed method on the previously existing knowledge, therefore in the quantitative evaluation we can not asses, if the method has managed to discover some potentially useful and previously undiscovered knowledge.

We are interested in precision at rank k. This means that only the candidates ranked equal to or higher than k are considered and the rest are disregarded. Precision is the ratio of the number of relevant candidates divided by the number of candidates returned by the system, or more formally:

$$\text{precision} = \frac{|\text{relevant candidates@}k|}{|\text{returned candidates}|}$$

Recall@k is the ratio of the number of relevant candidates ranked equal to or higher than k by the system divided by the number of correct ground truth candidates:

$$\text{recall} = \frac{|\text{relevant candidates@}k|}{|\text{correct candidates}|}$$

We measure precision and recall at k=100 and k=2802 in order to investigate how different number of retrieved candidates for each seed concept set affects the precision and recall of the methods. More specifically, we are trying to confirm or deny a hypothesis that larger k values degrade the overall precision of the method.

The relevance of the candidate is determined according to two matching criteria. First one is the **exact** match, where the candidate is deemed relevant

[10] The normalized Levenshtein difference threshold of 0.7 was chosen empirically.

[11] Note that the study by [5] is not included in the CORD-19 corpus used for training the embeddings, since it was published on May 14th 2020 and we use the CORD-19 version published on May 1st 2020.

if it appears in the list of identified targets in the study by [5]. The second is the **fuzzy** match, where we check if the targets belong to the same "family" as a specific confirmed target. This strategy was proposed by the medical experts and checks whether the prefix of the specific gene (characters in the gene name that appear before the first digit in the name) matches a prefix of a specific gene name in the list. We enforce an additional constraint that the matching prefixes need to be at least three characters long for a successful match in order to minimize the false positive rate.

Qualitative Evaluation We generated two distinct therapy target candidate lists using the proposed SciBERT based embedding method. First one contained 100 closest neighbours to the protein ACE2 according to the cosine distance between embeddings, and the second one contained 100 closest neighbours to the protein TMPRSS2. Both lists were given to the medical expert who inspected the list for possible previously undiscovered candidates.

6 Results

Here we present the results of the quantitative and qualitative evaluation.

6.1 Results of the quantitative evaluation

The results of the quantitative evaluation are presented in Table 2. In column ACE2 + TMPRSS2 we present results when these two proteins are used as seed concepts, and in column UBA2 + NCKAP1 we present results when these two proteins, which were chosen according to the largest value of the Ratio 24h criterion (see Section 5.3) are used as seed concepts. Left part of the Table presents results for the proposed approach based on SciBERT and the right part of the Table presents results for the baseline FastText approach in terms of precision and recall at two distinct k values (100 and 2802). EXACT indicates that exact matching is used and FUZZY indicates fuzzy matching (see Section 5.4).

Table 2. Results (precision@k and recall@k) of the quantitative evaluation for two seeds by the expert and two seeds from the literature. Best result in each row is bolded.

	SciBERT		**FastText**	
	ACE2 + TMPRSS2	UBA2 + NCKAP1	ACE2 + TMPRSS2	UBA2 + NCKAP1
EXACT P@100	0.110	**0.220**	0.040	0.170
EXACT R@100	0.004	**0.008**	0.001	0.006
EXACT P@2802	0.097	**0.118**	0.025	0.076
EXACT R@2802	0.097	**0.118**	0.025	0.076
FUZZY P@100	0.290	**0.490**	0.070	0.380
FUZZY R@100	0.010	**0.017**	0.002	0.014
FUZZY P@2802	0.222	**0.252**	0.092	0.183
FUZZY R@2802	0.222	**0.252**	0.092	0.183

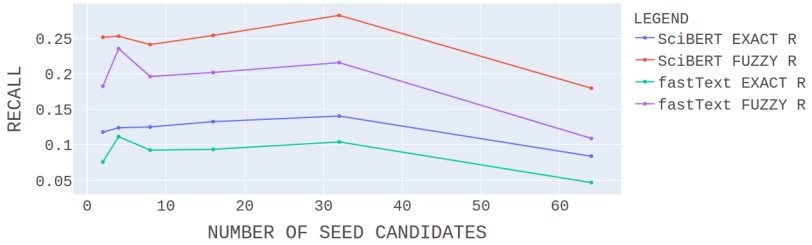


Fig. 2. Relation between recall and the number of seed candidates.

SciBERT based method outperforms the FastText baseline by a large margin in both seed therapy target acquisition scenarios and according to all the criteria. Using UBA2 + NCKAP1 works better than using ACE2 + TMPRSS2, achieving the best fuzzy precision@100 of 0.490 and the best exact precision@100 of 0.220. FastText baseline also works fairly well in this scenario, achieving fuzzy precision@100 of 0.380 and the best exact precision@100 of 0.170. When more (2802) candidates are obtained, the recall increases for both methods but at an expense of a significant drop in precision for both methods and for almost all configurations. The only exception is the increase in fuzzy precision by about 2 percentage points when FastText method and ACE2 + TMPRSS2 seed concepts are used. The most likely reason for the drop is that at larger k values some of the target candidates acquired by the method might be semantically too dissimilar to the seed targets, since more candidates per each seed therapy target need to be acquired in order to get the required amount of semantic neighbours (e.g., for k=2802, we get about 1401 semantic neighbours for each of the seed genes).

This raises the question of how many seed terms should be supplied to the system for the best performance when a large number of target candidates is required as output. Figure 2 shows the relation between the achieved recall@2802 (exact and fuzzy) of both methods when we increase the number of seed candidates (see Section 5.3 for details about our sampling procedure). For SciBERT based method, the best fuzzy and exact recalls are achieved when 32 seed candidates are used (28.2% and 14.1% respectively). On the other hand, the FastText based method shows a spike in performance when 4 seed concepts are used. This indicates that for some reason the two seed candidates ranked third and fourth (ENO1 and ATP5O, respectively) according to the Ratio 24 criterion have a very positive effect on the FastText model but not SciBERT. While we do not have a clear explanation for this phenomenon, it is hypothesized that it might be connected with morphological similarity between these two genes and other genes in the list of candidates proposed by [5], since FastText can also leverage morphological similarity. Spikes asides, the general trend for both methods and both recalls is quite similar. There is a gradual increase in performance for up to 32 seed candidates and after that the performance decreases.

Table 3. Genes/proteins (in alphabetical order) which are common to the TMPRSS2 and ACE2 list and their (putative) relevance to COVID-19.

Gene	Protein	Relevance to COVID-19
ATP2B2 (PMCA2)	Plasma membrane Ca^{2+}-transporting ATPase	?
CRTH2 (PTGDR2)	Prostaglandin D2 (PGD2) receptor	PGD2 is important for survival of mice infected with neurotropic coronavirus. Increased production of PGD2 is linked to increased mortality in aged mice. PGD2blockade improves survival in mice infected with SARS-CoV [35,37].
DPP7 (DPP2)	Dipeptidyl peptidase 2	DPP7 is associated with the magnitude of the antibody response to influenza vaccination [15].
MECP2	Methyl-CpG-binding protein 2	MECP2 duplication in humans is associated with IgA/IgG2 antibody deficiency and severe infections. Mice overexpressing MECP2 are hypersensitive to influenza A virus [1,10].
METAP2 (P67EIF2)	Methionine aminopeptidase 2 (Initiation factor 2-associated 67 kDa glycoprotein)	Plays a role in regulation of protein synthesis during vaccinia virus infection [6].
PLA2R1	Secretory phospholipase A2 (PLA2) receptor	Restricted activity of PLA2 is associated with improved survival in mice infected with HCoV-OC43. Inhibition of cytosolic PLA2 suppresses replication of HCoV-229E [13,27].
PTGS2 (COX2)	Prostaglandin G/H synthase 2 (cyclooxygenase-2)	SARS-CoV induces cyclooxygenase 2 [23].
SOX2	Transcription factor SOX-2	SOX2+ cells are important for regeneration of airway epithelium after severe influenza infection in mice [32].
SSTR2 (SST2)	Somatostatin receptor type 2	?

6.2 Results of the qualitative evaluation

Nine genes/proteins were the same in the ACE2 and TMPRSS2 lists, indicating they might be important for pathogenesis of COVID-19. The role of these genes/proteins in pathogenesis of COVID-19 has not been established, but indirect evidence supports this notion at least for some of them. Indeed, most of these genes/proteins have been previously linked to viral diseases, including those caused by SARS-CoV (a virus, which causes SARS, and is related to SARS-CoV-2), and other coronaviruses (Table 3). Furthermore, METAP2 and DPP7, which we identified as potentially relevant for COVID-19, were altered in cells infected with SARS-CoV-2, although the difference for DPP7 did not reach the level of statistical significance [5].

Interestingly, three proteins in Table 3 (PTGS2, CRTH2, and PLA2R1) are linked to infection with coronaviruses as well as metabolism of phospholipids and/or prostaglandin synthesis and action. Furthermore, both the ACE2 and TMPRSS2 lists contain genes/proteins, such as PLA2 (phospholipase A2, PLA2G2D (Group IID secretory phospholipase A2), and SPLA2 (secretory PLA2), which do not match directly, but are involved in the same or related cellular processes. Notably, increased expression of *Pla2g2d* in older mice was shown to be linked with increased mortality due to SARS-CoV infection [36]. In addition, a recent proteomic analysis has demonstrated that protein abundance of PLAA (phospholipase A2-activating protein), PLA2G4A (cytosolic phospholipase A2), and PLA2G2 (Group IIA phospholipase A2) is altered in cultured cells infected with SARS-CoV-2 [5], which gives further credence to the idea that phospholipid metabolism is important under these conditions. In summary, taken together with published experimental data, our analysis suggests that phospholipases and/or prostaglandins might represent a target for treatment of COVID-19.

7 Conclusions and further work

In this paper we presented a method for discovering new COVID-19 therapy targets by leveraging contextual embeddings, which outperforms the method based on FastText embeddings. We explored the best tactics for acquiring seed targets from the related work if expert knowledge is not available. The results of the manual qualitative evaluation by the expert indicate that at least two groups of novel therapy target candidates have been discovered.

The proposed method outperforms the baseline FastText method by a large margin, which can be explained by the fact that SciBERT is also leveraging knowledge gained during the pretraining on the large corpus of scientific literature, which enables the model to generate vector representations that reflect this wider semantic context. The drawback is however the difference in the amount of computational resources required by the two methods. We also acknowledge that the proposed method, which constructs static embeddings from the SciBERT contextual embeddings is not the only possibility for construction of meaningful semantic representations. Other possibilities and models (e.g., BioBERT [21]) will be explored in the future work. The quantitative evaluation indicates that the precision and recall of the method are still relatively low in most cases. This can on one side indicate that COVID-19 topic is not researched enough to confirm relations between COVID-19 and some candidates found by the proposed method. Another indication of this is the qualitative study, which confirmed that some of the proposed candidates found by the system have research potential but have not yet been explicitly confirmed as being related to COVID-19 in the existing literature.

On the other hand, low precision most likely also indicates that there is still a large amount of proposed candidates, which play no role in the advancement and prevention of the COVID-19 disease. Some of these false positives can be attributed to inadequate synonym resolution since the list used for that task (see Section 5.2) most likely covers only a small percentage of genes and their synonyms found in the CORD-19 corpus. Other mistakes can be contributed to the byte pair encoding scheme SciBERT employs. Since the model generates embeddings for subword tokens instead for an entire words (see how we deal with this problem in Section 5.1), some words with similar roots or affixes can perhaps appear closer in the semantic space as they should according to their semantic relatedness because of the morphological resemblance. We will address this issues in the future work.

Acknowledgements This work was funded by the Slovenian Research Agency (ARRS) through core research programme *Knowledge Technologies* (P2-0103), research project *Semantic Data Mining for Linked Open Data* (financed under the ERC Complementary Scheme, N2-0078), and a young researcher grant (BŠ). The work was also supported by EU Horizon 2020 research and innovation programme under grant agreement No 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media). The publication reflects only the authors' views and the EC is not responsible for any use that may be made of the information it contains.

References

1. Bauer, M., Kölsch, U., Krüger, R., Unterwalder, N., Hameister, K., Kaiser, F.M., Vignoli, A., Rossi, R., Botella, M.P., Budisteanu, M., et al.: Infectious and immunologic phenotype of mecp2 duplication syndrome. Journal of Clinical Immunology **35**(2), 168–181 (2015)
2. Beigel, J.H., Tomashek, K.M., Dodd, L.E., Mehta, A.K., Zingman, B.S., Kalil, A.C., Hohmann, E., Chu, H.Y., Luetkemeyer, A., Kline, S., et al.: Remdesivir for the treatment of covid-19—preliminary report. New England Journal of Medicine (2020)
3. Beltagy, I., Cohan, A., Lo, K.: Scibert: Pretrained contextualized embeddings for scientific text. arXiv preprint arXiv:1903.10676 (2019)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (Dec 2017)
5. Bojkova, D., Klann, K., Koch, B., Widera, M., Krause, D., Ciesek, S., Cinatl, J., Münch, C.: Proteomics of sars-cov-2-infected host cells reveals therapy targets. Nature pp. 1–8 (May 2020). https://doi.org/10.1038/s41586-020-2332-7
6. Bose, A., Saha, D., Gupta, N.K.: Viral infection: I. Regulation of protein synthesis during vaccinia viral infection of animal cells. Archives of Biochemistry and Biophysics **342**(2), 362–372 (1997)
7. Boulware, D.R., Pullen, M.F., Bangdiwala, A.S., Pastick, K.A., Lofgren, S.M., Okafor, E.C., Skipper, C.P., Nascene, A.A., Nicol, M.R., Abassi, M., et al.: A randomized trial of hydroxychloroquine as postexposure prophylaxis for covid-19. New England Journal of Medicine (2020)
8. Bruza, P., Weeber, M.: Literature-based Discovery. Springer Science & Business Media (2008)
9. Crichton, G., Baker, S., Guo, Y., Korhonen, A.: Neural networks for open and closed literature-based discovery. PLOS ONE **15**(5), 1–16 (05 2020)
10. Cronk, J.C., Herz, J., Kim, T.S., Louveau, A., Moser, E.K., Sharma, A.K., Smirnov, I., Tung, K.S., Braciale, T.J., Kipnis, J.: Influenza a induces dysfunctional immunity and death in mecp2-overexpressing mice. JCI Insight **2**(2) (2017)
11. Deftereos, S.N., Andronis, C., Friedla, E.J., Persidis, A., Persidis, A.: Drug repurposing and adverse event prediction using high-throughput literature analysis. Wiley Interdisc. Reviews: Systems Biology and Medicine **3**(3), 323–334 (2011)
12. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint: 1810.04805 (2018)
13. Do Carmo, S., Jacomy, H., Talbot, P.J., Rassart, E.: Neuroprotective effect of apolipoprotein d against human coronavirus oc43-induced encephalitis in mice. Journal of Neuroscience **28**(41), 10330–10338 (2008)
14. Grujicic, D., Radevski, G., Tuytelaars, T., Blaschko, M.B.: Self-supervised context-aware covid-19 document exploration through atlas grounding (2020)
15. HIPC-I Consortium, et al.: Multicohort analysis reveals baseline transcriptional predictors of influenza vaccination responses. Science immunology **2**(14) (2017)
16. Hoffmann, M., Kleine-Weber, H., Schroeder, S., Krüger, N., Herrler, T., Erichsen, S., Schiergens, T.S., Herrler, G., Wu, N.H., Nitsche, A., et al.: Sars-cov-2 cell entry depends on ace2 and tmprss2 and is blocked by a clinically proven protease inhibitor. Cell (2020)
17. Hope, T., Portenoy, J., Vasan, K., Borchardt, J., Horvitz, E., Weld, D.S., Hearst, M.A., West, J.: Scisight: Combining faceted navigation and research group detection for covid-19 exploratory scientific search. arXiv preprint: 2005.12668 (2020)

18. Korhonen, A., Guo, Y., Baker, S., Yetisgen-Yildiz, M., Stenius, U., Narita, M., Liò, P.: Improving literature-based discovery with advanced text mining. In: DI Serio, C., Liò, P., Nonis, A., Tagliaferri, R. (eds.) Computational Intelligence Methods for Bioinformatics and Biostatistics. pp. 89–98. Springer, Cham (2015)
19. Kostoff, R.N., Briggs, M.B., Lyons, T.J.: Literature-related discovery (LRD): Potential treatments for multiple sclerosis. Technological Forecasting and Social Change **75**(2), 239–255 (2008)
20. Kudo, T., Richardson, J.: Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. arXiv preprint:1808.06226 (2018)
21. Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C.H., Kang, J.: BioBERT: a pre-trained biomedical language representation model for biomedical text mining. Bioinformatics **36**(4), 1234–1240 (2020)
22. Liu, J., Cao, R., Xu, M., Wang, X., Zhang, H., Hu, H., Li, Y., Hu, Z., Zhong, W., Wang, M.: Hydroxychloroquine, a less toxic derivative of chloroquine, is effective in inhibiting sars-cov-2 infection in vitro. Cell Discovery **6**(1), 1–4 (2020)
23. Liu, M., Gu, C., Wu, J., Zhu, Y.: Amino acids 1 to 422 of the spike protein of sars associated coronavirus are required for induction of cyclooxygenase-2. Virus genes **33**(3), 309–317 (2006)
24. Martinc, M., Novak, P.K., Pollak, S.: Leveraging contextual embeddings for detecting diachronic semantic shift. arXiv preprint arXiv:1912.01072 (2019)
25. Mehra, M.R., Desai, S.S., Kuy, S., Henry, T.D., Patel, A.N.: Retraction: Cardiovascular disease, drug therapy, and mortality in covid-19. New England Journal of Medicine (2020)
26. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
27. Müller, C., Hardt, M., Schwudke, D., Neuman, B.W., Pleschka, S., Ziebuhr, J.: Inhibition of cytosolic phospholipase a2α impairs an early step of coronavirus replication in cell culture. Journal of virology **92**(4), e01463–17 (2018)
28. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Proc. of the 2014 conference on empirical methods in natural language processing (EMNLP). pp. 1532–1543 (2014)
29. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint:1802.05365 (2018)
30. Povey, S., Lovering, R., Bruford, E., Wright, M., Lush, M., Wain, H.: The hugo gene nomenclature committee (hgnc). Human genetics **109**(6), 678–680 (2001)
31. Pyysalo, S., Baker, S., Ali, I., Haselwimmer, S., Shah, T., Young, A., Guo, Y., Högberg, J., Stenius, U., Narita, M., Korhonen, A.: LION LBD: a literature-based discovery system for cancer biology. Bioinformatics **35**(9), 1553–1561 (10 2018)
32. Ray, S., Chiba, N., Yao, C., Guan, X., McConnell, A.M., Brockway, B., Que, L., McQualter, J.L., Stripp, B.R.: Rare sox2+ airway progenitor cells generate krt5+ cells that repopulate damaged alveolar parenchyma following influenza virus infection. Stem cell reports **7**(5), 817–825 (2016)
33. Swanson, D.R.: Medical literature as a potential source of new knowledge. Bulletin of the Medical Library Association **78**(1), 29 (1990)
34. Tshitoyan, V., Dagdelen, J., Weston, L., Dunn, A., Rong, Z., Kononova, O., Persson, K.A., Ceder, G., Jain, A.: Unsupervised word embeddings capture latent knowledge from materials science literature. Nature **571**(7763), 95–98 (2019)

35. Vijay, R., Fehr, A.R., Janowski, A.M., Athmer, J., Wheeler, D.L., Grunewald, M., Sompallae, R., Kurup, S.P., Meyerholz, D.K., Sutterwala, F.S., et al.: Virus-induced inflammasome activation is suppressed by prostaglandin d2/dp1 signaling. Proceedings of the National Academy of Sciences **114**(27), E5444–E5453 (2017)
36. Vijay, R., Hua, X., Meyerholz, D.K., Miki, Y., Yamamoto, K., Gelb, M., Murakami, M., Perlman, S.: Critical role of phospholipase a2 group iid in age-related susceptibility to severe acute respiratory syndrome–cov infection. Journal of Experimental Medicine **212**(11), 1851–1868 (2015)
37. Zhao, J., Zhao, J., Legge, K., Perlman, S.: Age-related increases in pgd 2 expression impair respiratory dc migration, resulting in diminished t cell responses upon respiratory virus infection in mice. The Journal of clinical investigation **121**(12), 4921–4930 (2011)
38. Zhou, P., Yang, X.L., Wang, X.G., Hu, B., Zhang, L., Zhang, W., Si, H.R., Zhu, Y., Li, B., Huang, C.L., et al.: A pneumonia outbreak associated with a new coronavirus of probable bat origin. nature **579**(7798), 270–273 (2020)
39. Zhu, N., Zhang, D., Wang, W., Li, X., Yang, B., Song, J., Zhao, X., Huang, B., Shi, W., Lu, R., et al.: A novel coronavirus from patients with pneumonia in china, 2019. New England Journal of Medicine (2020)

Appendix D: A bilingual approach to specialised adjectives through word embeddings in the karstology domain

A bilingual approach to specialised adjectives through word embeddings in the karstology domain

(The paper will be published as part of TOTH 2020 proceedings http://toth.condillac.org/proceedings)

Larisa Grčić Simeunović[1], Matej Martinc[2], Špela Vintar[3]

University of Zadar[1], Jožef Stefan Institute [2], University of Ljubljana[3]

M. Pavlinovića 1, HR-23000 Zadar[1], Jamova cesta 39, SI-1000 Ljubljana[2], Aškerčeva 2, SI-1000 Ljubljana[3]

lgrcic@unizd.hr, matej.martinc@ijs.si, spela.vintar@ff.uni-lj.si

Abstract

We present an experiment in extracting adjectives which express a specific semantic relation using word embeddings. The results of the experiment are then thoroughly analysed and categorised into groups of adjectives exhibiting formal or semantic similarity. The experiment and analysis are performed for English and Croatian in the domain of karstology using data sets and methods developed in the TermFrame project. The main original contributions of the article are twofold: firstly, proposing a new and promising method of extracting semantically related words relevant for terminology, and secondly, providing a detailed evaluation of the output so that we gain a better understanding of the domain-specific semantic structures on the one hand and the types of similarities extracted by word embeddings on the other.

1 Introduction

In this paper we explore the bilingual comparative approach to adjectives through word embeddings in the domain of karstology. Because specialized adjectives have not received much attention in the field of terminology so far we aim at modelling the semantic relations expressed by adjectives within multiword terms and discovering their shared properties through word embeddings. Our starting hypothesis is that adjectival semantic relations are an important part of conceptual representations as they establish links between terms and their attributes inside conceptual frame. For this reason, we aim to reveal adjectives expressing exemplar characteristics in order to establish attribute-value pairs for different conceptual categories in karst domain.

By leveraging word embeddings and sets of seed adjectives expressing specific semantic relations we aim to extract additional adjectives that express the same relation and rate the degree of semantic similarity between adjectives in the two languages. Furthermore, we perform a detailed analysis of the extracted candidates in terms of their semantic and morphosyntactic properties in order to identify corresponding clusters between adjectives, but also to better understand the errors produced by our prediction algorithm.

Karstology is an interdisciplinary domain which studies karst, a type of landscape developing on soluble rocks such as limestone or gypsum. The most distinctive features of karst regions include caves, various types of relief depressions, submerged rivers, springs, ponors and sinkholes. The TermFrame project models the karstology domain using the frame-based approach (Faber, 2012; Faber, 2015). To explore typical conceptual frames in karstology we developed a domain-specific concept hierarchy of semantic categories, and each category can be described by a set of relations which reveal its specific features. In addition to manually identified categories and relations we employ a number of advanced text mining techniques to extract structured domain knowledge from our corpora (Miljković et al., 2019; Pollak et al., 2019; Vintar et al., 2020, Pollak et al. 2020). The final result of the project will be a multilingual visual knowledge base for karstology.

The experiment and analysis presented in this paper seek to highlight the synergies and parallels between the frame-based and cognition-inspired view of specialised language on the one hand, and on the other hand word embeddings as a mathematical device to map meaning into a multidimensional space. It is no coincidence that we perceive cognition as a dynamic and inherently spatial operation, and recent advances in deep learning for NLP prove that conceptual similarities are indeed reflected in the spatial proximity or closeness of word embeddings. By performing a detailed analysis of the adjectives extracted through embeddings we aim to shed light on the different nuances of semantic similarity as computed via deep learning methods.

The paper is structured as follows: In Section 2 we present related research, Section 3 describes the methods used for the extraction experiment and manual analysis, and Section 4 presents the analysis of adjectival clusters for each semantic relation and language respectively. We conclude with a discussion and final observations.

2 Related work

The representation of relations between concepts has already been presented in different terminological works which aim to illustrate the dynamics of cognition. From the point of view of Frame-Based Terminology (Faber et al. 2007; Faber and Leon Araúz 2016; Gil-Berrozpe et al. 2019; Cabezas-García and Leon Araúz 2018) knowledge structures are organized as frames on the basis of elements and entities which share similar contexts and situations. The relations between them allow us to construct meaningful schemes or mental representations of segments that belong to a particular specialized domain.

Even though previous work on exemplar models does not include explicit representation of attributes and their values, Barsalou (1992: 25) considers that recognising the values of the same attribute relies on the "embedded level of exemplar processing for categorizing characteristics as values of attribute categories ".

According to Petersen (2015), "[t]he attributes in a concept frame are the general properties or dimensions by which the respective concept is described"[1]. Their values express specifications that are important for creating concept frames or models for the representation of concepts. Such

[1] The attributes are defined by Barsalou (1992:30) as „a concept that describes an aspect of at least some category members. For example, color describes an aspect of birds. (…) A concept is an attribute only when it describes an aspect of a larger whole. When people consider color in isolation (e. g., thinking about their favorite color), it is not an attribute but is simply a concept".

frames can be used as templates to guide the formulation of definitions (Duran Muñoz, 2016), or they can be applied to predict the semantic category of an attribute in a multi-word term, as our study illustrates.

According to Baroni et al. (2014), context predicting models are a promising way for performing a number of experiments on different semantic similarity tasks and datasets such as semantic relatedness, synonym detection, concept categorisation, selectional preferences and analogy.

Diaz et al. (2016) and Pollak et al. (2019) who conducted previous research on set expansion tasks showed that embeddings can be successfully employed for query expansion on domain specific texts.

3 Methodology

For the purposes of our research, we use the English and Croatian part of the TermFrame corpus which contains relevant contemporary works on karstology and is representative in terms of the domain and text types included. It comprises scientific texts (scientific papers, books, articles, doctoral and master's theses, glossaries and dictionaries) from the field of karstology, which in itself is an interdisciplinary domain partly overlapping with surface and subsurface geomorphology, geology, hydrology and other fields. Table 1 gives basic information about the corpus.

	English	Croatian
Tokens	2,721,042	1,229,368
Words	2,195,982	969,735
Sentences	97,187	53,017
Documents	57	43

Table 1: Corpus information

In our previous research (Vintar et al., 2020) we proposed a method to extract expressions pertaining to a specific semantic relation from a comparable English and Croatian corpus by providing a limited number of seed words for each language and relation, then using intersections of word embeddings to identify words belonging to same relation class. An evaluation of the extracted candidates showed high variability in precision between relations and languages, ranging from 0.28 (FUNCTION, Croatian) to 0.80 (COMPOSITION, Croatian).

In this study we continue our analysis by exploring contexts where adjective-noun phrases are used to express semantic relations, by grouping the extracted adjectives into clusters according to their semantic and morphosyntactic properties, and by analysing erroneously extracted candidates. First, we classify the adjectives according to their semantic relation guided by the conceptual frame. The semantic relations of the adjectives are determined according to their dominant meaning in the domain of karst. For the purposes of this analysis adjectives are assigned to one of

the 5 semantic relations: LOCATION (*underground* cave), CAUSE (*fluvial* sediment), FORM (*vertical* shaft), COMPOSITION (*gypsum* karst) and FUNCTION (*soluble* rock).

Nevertheless, it is important to bear in mind that this classification is not unambiguous as adjectives can take different meanings depending on the nouns they modify. For example, in a phrase like *korozijski proces* the value *korozijski* is assigned to the relation FUNCTION, while in a syntagm *korozijska ponikva* the value of the adjective is CAUSE.

Secondly, we investigate the extraction of adjectives from word embeddings trained on English and Croatian specialised corpora. Specifically, the task is to find adjectives that qualify their respective headwords along 5 fixed semantic dimensions we refer to as relations, using a set of seed adjectives for each relation and language. We trained FastText embeddings (Bojanowski et al., 2017) on the English and Croatian part of the TermFrame corpus respectively, using the skip-gram model with the embedding dimension of 100. For each seed adjective expressing a specific semantic relation, we extract a set of 100 closest words according to the cosine distance. Then, we calculate intersections between these sets of closest words, for all combinations of seed adjectives and subset sizes 2 – 10 (see Vintar et al., 2020).

	location		function		form		composition		cause	
	en	cr	en	cr	en	cr	en	cr	en	cr
N	357	228	147	152	164	152	293	244	183	181
C	118	88	68	43	108	97	184	197	88	132
P	0.33	0.39	0.46	0.28	0.66	0.64	0.63	0.80	0.48	0.73

Table 2: Precision per semantic relation and language (N = number of extracted words, C = correct, P = precision)

The results show the overall lowest and highest precisions in both languages as well as large differences between individual semantic relations. The prediction of semantic class membership is confirmed even for expressions with very low frequency.

In Section 4 we present the results of a manual analysis of adjectives extracted through the embeddings intersection method. In particular, we look for clusters of morphosyntactically, semantically or derivationally related words and look for patterns of similarity across the two languages. To perform our analysis, we introduce suffix and prefix-based clusters as well as derivational cluster. The proposed method helps improve the interpretability of the embedding results.

4 Analysing patterns of similarity

In the following subsections we attempt to categorise the extracted adjectives into groups or clusters, whereby we consider the seed adjectives provided for each semantic relation prior to the extraction task. We present results for English and Croatian respectively, then a comparison is made between findings.

4.1 CAUSE

4.1.1 English

Seed adjectives: *allogenic, anthropogenic, fluvial, alluvial, erosional, solutional, periglacial, tectonic, volcanic, lacustrine, aeolian*

a) Suffix-based clusters corresponding seed words:

allogenic, anthropogenic: **-genic**: epigenic, geogenic, cryogenic, autogenic, orogenic, biogenic, pathogenic, hypogenic, glacigenic, monogenic, rheogenic, speleogenic, radiogenic, guanogenic

fluvial, alluvial: **-luvial**: eluvial, colluvial, pluvial, deluvial,

periglacial: **-glacial**: preglacial, subglacial, fluvioglacial, englacial, proglacial, supraglacial, postglacial, paraglacial, pleniglacial, glaciofluvial,

-al *(erosional, solutional)***:** disolutional, denudational, compressional, tensional, gravitational, lagoonal, formational, corrosional, depositional, torrential, detrital, deglacial, abrasional, suffosional, evolutional, dissolutional, subaerial

b) Suffix-based clusters not corresponding seed words:

-ous: terrigenous, autochthonous, calcareous, argillaceous, igneous

-clastic: thermoclastic, volcanoclastic, bioclastic, pyroclastic, clastic, siliclastic, siliciclastic

-karstic: glaciokarstic, fluviokarstic,

-genetic: paragenetic,

No group: lacustrine

4.1.2 Croatian

Seed adjectives: *alogen, antropogen, fluvijalan, erozijski, aluvijalan, vulkanski, lakustrijski, eolski, periglacijalni, tektonski*

a) Suffix-based clusters corresponding seed words:

alogen, antropogen: **-gen**: egzogen, kemogen, zoogen, biogen, kriogen, epigenijski, orogenski;

fluvijalan, aluvijalan: **-luvijalan** and **-fluvijalan**: iluvijalan, proluvijalan, delovijalan, diluvijalan, koluvijalan, glaciofluvijalan, postfluvijalan;

periglacijalan: **(-glacijalan)**: glacijalan, proglacijalan, interglacijalan, postglacijalan, fluvioglacijalan;

-ski *(erozijski, eolski)*: abrazijski, mikroerozijski, disolucijski, denudacijski, fluviodenudacijski, derazijski, destrukcijski, dislokacijski, soliflukcijski, subdukcijski, inundacijski, marinski, međuzrnski, siparski, eforacijski, amfibolski, supsidencijski, egzarazijski, padinski, mindelski, evolucijski, monoklinski, piraterijski, magmatski, evorzijski, melioracijski, kriofrakcijski,

translacijski, poligonski, riski, oligocenski, intrabazenski, plutonski, drobinski, akumulacijski, superpozicijski, litotamnijski, submarinski, regresijski, alveolinski, osulinski

-an, -ni: erozivan, abrazivan, piroklastični, naplavni, terasan, stadijalan, riječni, šljunčani, žilni, bazalni, bazaltni, pretaložen, denudiran, hipoabisalan, nataložen, bujičan, naplavljen, bujičav, pleistocenalan;

b) Suffix-based clusters not corresponding seed words:

-čan : klastičan, vulkanoklastičan, piroklastičan;

-genetski : biogenetski, klimamorfogenetski, tektogenetski, epigenetski, poligenetski, epirogenetski;

Observations:

Starting from the seed words in two languages we can observe high correspondence in results for this relation. Clusters around the same suffixes are formed in both languages:

-genic *(allogenic, anthropogenic)* (14) and ***-gen*** *(alogeni, antropogeni)* (7),

-luvial *(fluvial, alluvial)* (4) and ***-luvijalni*** *(fluvijalni, aluvijalni)* (8)

-glacial *(periglacial)*(10) *and* ***-glacijalni*** *(periglacijalni)*(5)

The remainder of the results also exhibit productive suffixes that can be specific for this semantic relation: the suffixes **-al, -ous, -clastic**, **-karstic, -genetic** in English and the suffixes **-ski, -an, -ni, -čan, -genetski** in Croatian.

This confirms that embedding methodology successfully retrieves adjectives from the same semantic relation.

4.2 COMPOSITION

4.2.1 English

Seed adjectives: *carbonate, limestone, dolomitic, sedimentary, sulphate, calcareous, carboniferous, silicate, sulphuric, diagenetic, siliceous, clay*

a) Suffix-based clusters corresponding seed words:

carbonate: metacarbonate, noncarbonate, bicarbonate;

limestone: **-stone:** rimestone, grainstone, dolostone, framestone;

dolomitic, sulfuric, diagenetic: **-ic:** evaporitic, quartzitic, conglomeratic, loessic, calcitic, sulphuric, silicic, pelitic, andesitic, carbonatic, carbonic, bioclastic, granitic, basaltic, magmatic, metallic, elastic, aphanitic, ophiolitic;

calcareous, carboniferous, siliceous: **-ous:** gypsiferous, argillaceous, igneous, amorphous, herbaceous, carboniferous, siliceous, carbonaceous, fossiliferous, sulfurous, tufaceous;

silicate, sulfate: **-ate**: vanadate, cipitate, dehydrate;

b) Suffix-based clusters not corresponding seed words:

-ite: siderite, anhydrite, hexahydrite, bassanite, phyllite, kaolinite, tyuyamunite, biomicrite, biopelmicrite, kimberlite, phosphorite, micrite, halloysite, serpentinite, dickite, alterite, barite, laterite;

-**mineral**: thermomineral, monominerallic;

-**grained:** finegrained;

No group: shale, schist, sandy, oxidase, paleokarstic, sulfide, silty, zechstein, claypan, foraminifer, manganese, haematite, chalky, nonsoluble, interclas, silicify, phyllito, monocrystalline, oxide,

4.2.2 Croatian

Seed adjectives: *karbonatan, vapnenački, dolomitski, sedimentan, kalcitan, karbonski, sulfatan, glinovit, sedren, stijenski*

a) Suffix-based clusters corresponding seed words:

karbonatan: nekarbonatan, hidrokarbonatan;

kalcitan, sedimentan, sulfatan: **-an**: aragonitan, detritičan, pješčan, kristaličan, rudistan, silikatan, evaporitan, gipsan, flišan;

dolomitski, karbonski, stijenski: **-ski:** alveolinski, amfibolski, drobinski, foraminiferski, morenski;

vapnenački: **-čki**: škriljevački;

glinovit: **-it:** laporovit, muljevit, pjeskovit, šljunkovit*;*

b) Suffix-based clusters not corresponding seed words:

zrnat: krupnozrnat, međuzrnski, sitnozrnat, zrnat/ zrnast;

Observations:

The embedding candidates in English can be organised according to productive suffixes like **-ic, -ous, -ate, -clastic, -ite, -mineral, -grained.**

In Croatian suffixes **-an/-ni, -ski, -čki, -it, -zrnat** are productive for the semantic relation composition.

4.3 FORM

4.3.1 English

Seed adjectives: *polygonal, vertical, dendritic, shallow, enclosed, elongated, flat, steep, cavernicolous, detrital*

a) Suffix-based clusters corresponding seed words:

polygonal, vertical: **-al**: elliptical, elliptic, cylindrical, subparallel, subhorizontal, subvertical, centripetal, symmetrical, subhorizontal, sinusoidal, asymmetrical, orthogonal, rectilinear, concave, angular, rectangular;

cavernicolous: **-ous**: sinuous;

No group: staircase, meander, singular, elbow, cylinder, labyrinthine, sharply, pinnacle, steeply,

4.3.2 Croatian

Seed adjectives: *vertikalan, ravnocrtan, strm, kavernozan, horizontalan, mrežast, longitudinalan, kružan, razgranat, ulegnut, uravnjen*

a) Suffix-based clusters corresponding seed words:

vertikalan, ravnocrtan, horizontalan, longitudinalan: **-an**: konveksan, tangencijalan, trodimenzionalan, subhorizontalan, simetričan, asimetričan, paralelan, tlocrtan, konkavan, ustrmljen, ovalan, nepravilan, polukružan, vodoravan, cilindričan, centrifugalan, centripetalan, konusan, dijagonalan, lateralan, longitudinalan, horizontalan, radijalan, konvergentan, etažan, urušan;

mrežast: **-ast**: ravničast, stepeničast, zvjezdast, žljebast, grozdast, prstenast, rešetkast, laktast, klifast, bunarast, dolinast, ponikvast, sigast, stepeničast, terasast;

b) Derivational cluster

Adjectives like *kavernozan, prevjesan, sinklinalan, monoklinalan, abisalan, fleksuran* are derived from the terms denoting karst forms:

kavernozan<kaverna, prevjesan<prevjes, sinklinalan<sinklinala, monoklinalan<monoklinala, abisalan<abis, fleksuran<fleksura;

No group: valovit, meandrirajući.

Observations: The results in this semantic relation exhibit clusters around seed adjectives *vertical, polygonal* in English and *vertikalan, ravnocrtan, horizontalan, longitudinalan* in Croatian. These adjectives refer to different shapes that are frequent in karst relief but can also refer to different other entities in nature.

Results in Croatian demonstrate a group of relational adjectives derived from karst forms: kavernozan<kaverna, prevjesan<prevjes, sinklinalan<sinklinala, monoklinalan<monoklinala, abisalan<abis, fleksuran<fleksura.

The suffixe **-ast** is also productive for expressing karst shapes in Croatian.

We can observe that the embeddings missed to propose adjectives that refer to karst forms. For. ex. *bunarast, ponikvast, terasast, krovinski, sigasti, zaravnjen, dolinast, škrapski*

4.4 FUNCTION

4.4.1 English

Seed adjectives: *impermeable, permeable, solutional, hydrothermal, speleological, geological, soluble, porous, depositional, regressive, undersaturated*

a) Suffix-based clusters corresponding seed words:

soluble: **-ble**: insoluble, impenetrable, rechargeable;

porous: **-ous**: anhydrous, impervious;

speleological, geological: **-logical**: paleontological, speleological, seismological, speleogical, petrological, biospeleological, climatological, vulcanospeleological, sedimentological, karstological;

Results ending in **-logical** but not connected to the karst domain: immunological, archeological, meteorological,histological, palynological, psychological, methodological, mythological,

No group: unkarstify, evaporitic, aquifer, dissociate, unsaturate, unconformable, lithoclast, friable, diffuse;

4.4.2 Croatian

Seed adjectives: *nepropustan, propustan, speleološki, geološki, topiv, porozan, taložan, urušan*

a) Suffix-based clusters corresponding seed words:

topiv: **-iv:** netopiv, vodotopiv, vododrživ, vododržljiv, propustljiv;

porozan, taložan, urušan, nepropustan, propustan: **-an:** vodonepropustan, vodopropusan, vodoprohodan, nepropusan, polupropusan, laminaran, difuzan, procjedan;

speleološki, geološki: **-ški:** speleomorfološki, geomofološki, etnološki, geoekološki, arheološki, aerološki, fiziološki, geoekološki, geokronološki, biološki, paleontološki,

No group: kemogen

Observations:

Seed adjectives *soluble, porous* in English and *topiv, porozan* in Croatian create suffix-based clusters in **-ble**, **-ous** in English and in **-iv**, **-an** in Croatian. While the results in English missed to propose the adjective *permeable*, results in Croatian demonstrate productive cluster around the seed adjectives *propustan, nepropustan.*

The suffixes **-logical, -loški** are productive in both languages since they refer to general concepts such as speleology, karstology, paleontology, seismology, slimatology, sedimentology.

4.5. LOCATION

4.5.1 English

Seed adjectives: *coastal, littoral, sublittoral, submarine, oceanic, subsurface, subterranean, subterraneous, subaerial, underground, aquatic, subaqueous, internal, subglacial, phreatic, epiphreatic, vadose*

a) Suffix-based clusters corresponding seed words:

coastal, littoral: **-al**: sublittoral, paralittoral, abyssal, intracontinental, continental, peripheral;

phreatic, epiphreatic: **-ic:** bathyphreatic;

b) Suffix-based clusters not corresponding seed words:

-ic: meteoric, aquatic, semiaquatic, atlantic, pacific, anastomotic;

-al: interfluvial, terrestrial, superficial;

-most: uppermost, lowermost;

-flow: underflow, downflow;

-haline: polyhaline, sakhalin, euhaline, anchihaline, mixohaline;

-shore: seashore, offshore;

c) Prefix-based clusters corresponding seed words:

subterraneous, subaqueous, subaerial, submarine, subsurface: **sub-:** subterranean, subvertical, subtidal, subhorizontal, subzone;

d) Prefix-based clusters not corresponding seed words:

sea-: seafloor, seawater;

hypo-: hyporheic, hypokarst, hyporheal, hyporheo;

hyper-: hyperkarst;

Other: cavernicole, microcavernicole, cavernicolous, anastomose, aquaticus, shallowwater, saline, waterline, shoreline, ocean, subjacent, branchwork, mesovoid, lacustrine, intrastratal, streamway, surfacedwelling, crevicular, supratidal, interstice, marine, nonmarine,

4.5.2 Croatian

Seed adjectives: *obalan, litoralan, priobalan, podmorski, oceanski, podzeman, vadozan, podvodan, dolinski, špiljski, freatski, epifreatski*

a) Suffix-based clusters corresponding seed words:

obalan, litoralan, priobalan, vadozan, podvodan: **-an:** maritiman, bazalan, lateralan, inversan, otočan, dugovalan, saturiran;

podmorski, oceanski, dolinski, špiljski, freatski, epifreatski: **-ski:** zonski, primorski, dubokomorski, prekomorski, plitkomorski, obalski, priobalski, kontinentski, litoralizacijski, piedmontski, dubinski, sifonski, jamski, zavalski, ekvatorski, ponorski, kanalski, vršinski,;

Many of the adjectives ending in **-ski** are derived from karst forms: jamski<jama, zavalski<zavala, ekvatorski<ekvator, ponorski<ponor, kanalski< kanal,

b) Prefix-based clusters not corresponding seed words:

sub-: submarinski, subfreatski, subaerski;

Observations: Results show that this relation is not as precise as the others because embedding candidates cannot be interpreted univocally as locations and great number of mistakes lead us to consider the multidimensionality of the adjectives. In the next section we explain this dilemma in more detail.

4.6 Adjectives assigned to the wrong relation

As the results above show, the method based on word embeddings can be of broad use in pattern recognition. However it is also important to bear in mind the mistakes that appear along. Most of the mistakes can be found within the semantic relations of LOCATION, FUNCTION and COMPOSITION.

4.6.1 LOCATION

The adjective candidates that evidently do not determine location are the following:

- genetic: *epigenetic, mesogenetic, eogenetic, paragenetic, diagenetic, telogenetic, speleogenetic, ontogenetic;*

-genic: *epigenic, rheogenic, geogenic, basaltic, elliptic, orogenic;*

-gean: *hypogean, aegean, epigean/epigene, endogean;*

These adjectives determine CAUSE and the embedding methodology extracted them as candidates. However, the error originates from the seed word selection where epigenic was wrongly assigned to the LOCATION relation. Two similar examples from the same group appear in Croatian list of candidates: *epigenetski, speleogenetski, poligenetski*.

English: The adjectives like *transversal, horizontal, vertical, elliptical, sinusoidal* cannot qualify location but FORM. The two examples from the same group also appear in Croatian list of candidates: *vodoravan, konveksan.*

The adjective *sulphuric* was recognised as qualifying LOCATION while it determines COMPOSITION.

Croatian: The adjectives like *vodotopiv, vododrživ, difuzan, porozan, hidrostatski, toplinski, drenažan, tlačni, nezasićen, procjedan, direktan, ozonski, protočan* do not signify LOCATION but FUNCTION.

4.6.2 FUNCTION

In this group we found examples that were recognised in two different semantic relations FUNCTION and COMPOSITION. This entails a need for further verification of the results in order to determine whether the adjectives show multidimensional meaning or they are erroneously attributed to a certain semantic relation. Examples in English and Croatian can confirm that their isolated meaning express only COMPOSITION. Nevertheless, it is important to pay attention to the nouns they modify which can explain the meaning shift.

English: *argillaceous, carbonaceous, siliceous, dolostone, limestone, ferruginous, gaseous, hydrous, cavernous*;

Croatian: *sitnozrnat/sitnozrnast, zrnat/zrnast, krupnozrnat/ krupnozrnast, sedimentan, glinovit karbonatan, karbonatni, hidrokarbonatan, nekarbonatan, dolomitan, dolomitičan, laporovit, gipsan.*

4.6.3. COMPOSITION

Embedding results for the semantic relation COMPOSITION also include adjectives that appear as candidates in two different semantic relations but only one is correct. For example, adjectives in Croatian like *vodonepropustan, vodopropusan, neotopiv, trošiv, polupropusni, topljiv, vododržljiv, vododrživ* appear in groups COMPOSITION and FUNCTION but only FUNCTION is valid.

Adjective *monoklinski* appeared as candidate in COMPOSITION and CAUSE but it only qualifies CAUSE.

The same goes for adjectives derived from the geographical eras like *staromezozojski*, *gornjokretacejski*, *postkretacejski* and *mladomezozojski*. Their meaning qualifies CAUSE and not COMPOSITION.

Mistakes in this group also include adjectives denoting FORM instead of COMPOSITION: *dendritičan, angularan, amorfan, sigasti, heksagonski*.

We can conclude that embedding results are able to point to polysemic meaning by categorising the same candidates in two different semantic groups. In these cases, a further validation will determine if only one meaning is correct or it is possible to describe the meaning as multidimensional. In the next paragraph we present several examples of adjectives whose meaning can be attributed to two different semantic relations depending on their definition or the nouns they are modifying.

4.7 Multidimensional adjectives

For certain adjectives we found that two relations are possible and that they should be described as multidimensional:

English:

carbonate, *metacarbonate*, *noncarbonate* (COMPOSITION AND FUNCTION)

evaporitic (COMPOSITION AND FUNCTION)

Croatian:

metamorfan, detritičan, neogenski, alogen, evaporitan: CAUSE and FUNCTION

kavernozan: LOCATION and FORM

pukotinski: LOCATION and FUNCTION

osulinski: LOCATION and CAUSE

Furthermore, we list adjectives which vary in meaning depending on their context. They appear in two or more relations but their meaning can only be confirmed after analysing the nouns they are usually combined with.

English:

igneous: COMPOSITION, FUNCTION, CAUSE

magmatic: COMPOSITION, FUNCTION, CAUSE, LOCATION

sediment: COMPOSITION, CAUSE,

subaerial: LOCATION, CAUSE, FUNCTION

siliciclastic: COMPOSITION, FUNCTION, CAUSE

Croatian:

međuzrnski: CAUSE, COMPOSITION AND LOCATION but only COMPOSITION is valid.

*klastičan, sitnoklastičan, kataklastičan, vulkanoklastičan***:** COMPOSITION, CAUSE, FUNCTION

The only example where embeddings did not suggest the right semantic relation are adjectives *gornjotrijaski, srednjotrijaski, trijaski, donjotrijaski* which were attributed relations COMPOSITION and FUNCTION but they refer to CAUSE.

5. Discussion and conclusions

We present a method for extracting semantically related adjectives using intersections of word embeddings and a detailed manual analysis of the extracted words. The results of the first stage show high variability in precision between relations, yet for three out of five target relations the method successfully extracts numerous meaningful adjectives pertaining to the target semantic relation.

The analysis performed in the second stage reveals several nuances of semantic similarity which we categorise into clusters. In most cases, members of a cluster share a surface linguistic component such as a suffix, prefix or word stem. Some suffixes indeed contain a semantic component pertaining to a specific relation (-genic -> CAUSE), and a shared word stem almost necessarily entails a similarity in meaning. In other cases, word embeddings allow us to retrieve synonyms with no surface similarity (*podmorski – submarinski*) only on the basis of their shared contexts.

While we cannot completely explain why a certain word is extracted as similar, we know that FastText embeddings combine the vector of the entire word and some of its substrings into a single vector. The cosine similarity thus entails also the similarity of the substrings, i.e. prefixes, suffixes and word stems.

In our future experiments we intend to extend the approach to longer expressions and multi-word terms, in particular structures which represent micro-events and which can potentially be modelled through embedding analogies.

Acknowledgements

The authors acknowledge the financial support from the Slovenian Research Agency for research core funding for the programme Knowledge Technologies (No.P2-0103) and the project TermFrame - Terminology and Knowledge Frames across Languages (No. J6-9372). This paper is also supported by European Union's Horizon 2020 research and innovation programme under

grant agreement No. 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media).

References

Barsalou, L. (1992). Frames, concepts, and conceptual Velds. In A. Lehrer & E. F. Kittay (Eds.), *Frames, Fields, and Contrasts: New Essays in Semantic and Lexical Organization*, Hillsdale, NJ: Lawrence Erlbaum Associates. pp. 21-74.

Cabezas-García, M. & León-Araúz, P. (2018) Towards the Inference of Semantic Relations in Complex Nominals: a Pilot Study. In Language Resources and Evaluation Conference (LREC 2018), edited by Calzolari, N., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Hasida, K., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S. & Tokunaga, T., pages 2511-2518. Miyazaki, Japan: ELRA.

Diaz, F., Bhaskar M., and Craswell, N. (2016). Query expansion with locally-trained word embeddings. *arXiv preprint arXiv:1605.07891*

Gil-Berrozpe, J.C., León-Araúz, P. & Faber, P. (2019) Ontological Knowledge Enhancement in EcoLexicon. In Proceedings of the eLex 2019 conference: Electronic lexicography in the 21st century, edited by Kosem, I., Zingano-Kuhn, T., Correia, M., Ferreira, J.P., Jansen, M., Pereira, I., Kallas, J., Jakubíček, M., Krek, S. & Tiberius, C., pages 177-197. Brno: Lexical Computing CZ, s.r.o.

Faber, P., León Araúz, P., Prieto Velasco, J.A. & Reimerink, A. (2007) Linking images and words: the description of specialized concepts. International Journal of Lexicography, 20(1):39-65. Oxford Univ Press.

Faber, P. & León-Araúz, P. (2016) Specialized knowledge representation and the parameterization of context. Frontiers in Psychology, 7 (00196).

Faber, P. (ed.) (2012). *A Cognitive Linguistics View of Terminology and Specialized Language*. Berlin, Boston: De Gruyter Mouton.

Faber, P. (2015) Frames as a framework in terminology. In Handbook of Terminology, edited by Kockaert, H.J. & Steurs, F., 1:14-33. John Benjamins Publishing Company.

Miljković D., Kralj, J., Stepisnik, U., and Pollak, S. (2019). Communities of related terms in a karst terminology co-occurrence network. *Proceedings of eLex19*, Sintra, Portugal. pp. 357-373.

Pollak, S., Repar, A., Martinc, M. and Podpecan, V. (2019). Karst exploration: extracting terms and definitions from karst domain corpus. *Proceedings of eLex19*, Sintra, Portugal. pp. 934-956.

Pollak, S., Podpečan, V., Miljkovic, D., Stepišnik, U. and Vintar, Š. (2020). The NetViz terminology visualization tool and the use cases in karstology domain modeling. In Proceedings of the 6th International Workshop on Computational Terminology (COMPUTERM 2020), pp. 55–61.

Vintar, Š., Saksida, A., Vrtovec, K., and Stepišnik, U. (2019). Modelling specialized knowledge with conceptual frames: the TermFrame approach to a structured visual domain representation. *Proceedings of eLex19*, Sintra, Portugal. pp. 305-318.

Vintar, Š., Grčić Simeunović, L., Martinc, M., Pollak, S. and Stepišnik, U. (2020). Mining Semantic Relations from Comparable Corpora through Intersections of Word Embeddings. In Proceedings

of the 13th Workshop on Building and Using Comparable Corpora, pp. 29–34. Language Resources and Evaluation Conference (LREC 2020).