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

D4.2: Initial multilingual news linking technology (T4.1)

Executive summary

Task *T4.1* aims to develop real-time multilingual news linking methods that are able to link news stories across languages based on different dimensions of the news content such as topics, events and entities. This deliverable describes the work to date on T4.1. We build on cross-lingual embeddings to develop method for efficient cross-lingual news linking. We also aim to support subsequent analysis of news collections as a whole, based on their contents. In this deliverable, we present several lines of work to these ends. We demonstrate *monolingual document linking* methods as a starting point and show the results of their application to monolingual collections. We also report on work on *news categorisation* and *language variety classification*, both important in supporting further analysis of the contents of a news collection. Then we report on work so far on *cross-lingual news linking*. We apply a number of different techniques, some novel and some replicated from literature, and find that a combination of methods performs best.

Partner in charge: UH

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:	**D4.2**
Deliverable full title:	**Initial multilingual news linking technology**
Deliverable short title:	**Initial multilingual linking technology**
Document identifier:	**EMBEDDIA-D42-InitialMultilingualLinkingTechnology-T41-submitted**
Lead partner short name:	**UH**
Report version:	**submitted**
Report submission date:	**30/06/2020**
Dissemination level:	**PU**
Nature:	**R = Report**
Lead author(s):	**Mark Granroth-Wilding (UH)**
Co-author(s):	**Elaine Zosa (UH), Lidia Pivovarova (UH), Matej Martinc (JSI), Vid Podpečan (JSI), Senja Pollak (JSI), Marko Pranjić (TRI)**
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
22/04/2020	v1.1	Mark Granroth-Wilding (UH)	Started draft report.
14/05/2020	v1.2	Elaine Zosa (UH)	Initial draft.
21/05/2020	v1.3	Vid Podpečan (JSI)	Article linking on Croatian datasets.
25/05/2020	v1.4	Matej Martinc, Senja Pollak (JSI)	SRNA, tax2vec and language variety sections.
25/05/2020	v1.5	Marko Pranjić (TRI)	Article linking on annotated triplets data.
26/05/2020	v1.6	Mark Granroth-Wilding, Elaine Zosa (UH)	Updated cross-lingual linking and reorganisation.
26/05/2020	v1.7	Senja Pollak (JSI)	More on monolingual sections.
27/05/2020	v1.8	Mark Granroth-Wilding (UH)	Finalised for internal review.
11/06/2020	v1.9	Marko Pranjić (TRI), Adrian Cabrera (ULR)	Internal review comments added.
22/06/2020	v1.10	All co-authors	Addressed comments from internal review.
24/06/2020	v2.0	Nada Lavrač (JSI)	Quality control.
25/06/2020	final	Mark Granroth-Wilding (UH)	Final changes post-QC.
30/06/2020	submitted	Tina Anžič (JSI)	Report submitted.

Table of Contents

List of abbreviations

BC	Betweenness Centrality
BERT	Bidirectional Encoder Representations from Transformers (word embedding method)
BON	Bag of n-grams
BOW	Bag of words
CLDR	Cross-Lingual Document Retrieval
CNN	Convolutional Neural Network, ConvNet
ExM	Ekspress Meedia
LDA	Latent Dirichlet Allocation (topic model)
LSA	Latent Semantic Analysis
LSI	Latent Semantic Indexing
LSTM	Long Short-Term Memory
mBERT	Multilingual BERT
MI	Mutual Information
MRR	Mean reciprocal rank
NLP	Natural Language Processing
PLTM	Polylingual Topic Model
PPR	Personalised PageRank
RNN	Recurrent Neural Network
SRNA	Semantics-Aware Recurrent Neural Architecture
STY	Styria Media Services
SVM	Support Vector Machine
TF-IDF	“term frequency - inverse document frequency” statistical measure
WSD	Word-sense disambiguation
XLM-R	XLM-RoBERTa (word embedding method)
YLE	Finland’s national broadcaster

1 Introduction

The overall objective of WP4, named *Cross-lingual content analysis*, is to facilitate the analysis of news content across different languages, aiming to empower news media consumers, researchers and news media professionals. The current language barriers and overflow of information prevent these groups from detecting and consuming all the relevant information, particularly across different languages, and from analysing and reflecting on the differences in news reporting. An important tool when working with large collections of news articles is *linking* – retrieving articles related in content or subject matter to a given article of interest. In a multilingual collection, this linking must be performed *cross-lingually*, to find articles that are not necessarily written in the same language as the query. WP4 aims to provide real-time linking of relevant texts with informative summaries, visualisations of content, as well as an analysis of the viewpoints and sentiment of articles from different sources, while addressing content in different languages.

This deliverable reports on the activities performed in Task 4.1 of WP4 of the EMBEDDIA project. In this task, *T4.1*, we address the problem of *news linking*, both monolingually and cross-lingually. News linking is the problem of finding closely related news articles in a corpus, for example, articles describing the same event or expressing opinions on the same issue. In a cross-lingual setting, the linked articles may be written in different languages. We aim to develop real-time multilingual news linking methods that are able to link news stories across languages based on different dimensions of the news content such as topics, events and entities. In this work, we combine cross-lingual embeddings from WP1 with input from WP2 (events, entities, etc.) to develop methods for efficient cross-lingual news linking.

Another goal in T4.1 is to support subsequent analysis of news as a whole, based on its contents. For instance, we aim at discovering what are the trending topics, which topics are of special interest in different countries, or how are the stances to a given event distributed in different countries. Our hypothesis is that combination of topic modelling techniques, cross-lingual embeddings, and other semantic enrichment methods should allow much richer access to and analysis of news stories across a multitude of languages.

In this deliverable, after presenting the available datasets (Section 2), we first present **monolingual** methods relevant to these aims, and demonstrate the application of a number of **news linking** methods to monolingual corpora to evaluate their effectiveness (Section 3). We then report on our work on **news categorisation** (Section 4), describing advances concerning the inclusion of semantic features and language variety classification, contributing the means for improved analysis of the contents of a news collection. Then, we report on our work so far on **cross-lingual news linking** (Section 5). We apply a number of different techniques, using cross-lingual embeddings and topic modelling, and compare them, finding that a combination of methods performs best. The report concludes with a list of associated outputs, conclusions, and the related papers included in Appendices A–E.

2 Input data

In this section, we describe the datasets used in the linking experiments reported below.

2.1 Croatian news dataset

The Croatian news dataset contains news articles from '24sata', the biggest Croatian news publisher. The dataset contains 546,801 articles published online between 2007-03-12 and 2019-04-24. Besides the articles in Croatian, the dataset contains the articles' metadata. Each entry contains the **title**, **lead_text**, **content** of the article, **author**, content **tags**, the **section** of the newspaper where the article appears, **published_date** (the date when the article was published on the website), **created_date**, (the date when the article was originally written), and **related_articles** containing a list of references to other articles. The *published_date* and *created_date* can differ as some articles are written in advance to be

published later (it should be noted that due to the redesign of the portal some of the dates were reset to the portal redesign date). The list of related articles is chosen by the journalist when the article is written and links to related articles are embedded in the content of the article when the article is published. The information about related articles is used in our evaluation as a ground truth and we use it to evaluate different text representation methods. The length of news articles in the datasets vary from 11 to 19,695 words, with an average length of 272 words.

2.2 YLE news dataset

The YLE dataset is composed of news articles from 2011 to 2018 in Finnish and 2012 to 2018 in Swedish, made available by Finland's national broadcaster, YLE. The articles are written separately and therefore the dataset does not represent a parallel corpus. This dataset is publicly available and can be downloaded from the Language Bank of Finland[1]. There are 604,297 Finnish articles and 228,473 Swedish articles. There are several metadata associated with each article. Notably, for our task, subjects (or keywords) are associated with each article. These includes the subjects (ranging from named entities to general concepts like sports and economy) discussed in the article. The subjects are assigned a unique identifier and have links to external databases such as Wikidata.

2.3 Estonian news dataset

Ekspress Meedia (ExM) is the leading media group in the Baltic States, whose activities include publishing, printing services, and online media content production. ExM owns the leading online media portals in the Baltics and publishes Estonia's most widely read daily and weekly newspapers, in addition to seven out of the top ten magazines in Estonia. We use a dataset from Ekspress Meedia that contains news articles in Estonian and Russian from digital editions of a number of different publications

The dataset was prepared by the ExM IT department. It is an archive of all publicly visible articles from Estonian and Russian news portals from the year 2009 to May 2019. The datasets can be used for research purposes by the researchers of the consortium without any specific limitations during the project and for research after the project.

2.4 Public datasets for news categorisation

In the approaches of news categorisation with background knowledge (SRNA and tax2vec), described in Section 4, we used public datasets with category labels.

The news datasets include:

- **Reuters data set**: consists of 11,263 newspaper articles, belonging to 46 different topics (classes).[2]
- **BBC news data set:** consists of 2225 documents attributed to five topic categories (business, entertainment, politics, sport, tech)[3] (Greene & Cunningham, 2006).

In the experiments we used also non-news dataset that we summarise with the purpose of understanding the reported experiments:

- **IMDB review data set**: consists of 50,000 reviews. Here, the goal is to predict the sentiment of individual reviews (positive or negative). The data set was obtained from the Keras library.
- **PAN reviews data set**: consists of reviews written by 4160 authors (2080 male and 2080 female). Reviews written by the same author are concatenated in a single document. The goal is to classify the author's gender. Detailed description of the data set is given in Rangel et al. (2014).

[1] https://www.kielipankki.fi/corpora/
[2] This data set is loaded via the Keras library (https://keras.io/datasets/)
[3] https://github.com/suraj-deshmukh/BBC-Dataset-News-Classification/blob/master/dataset/dataset.csv

- **PAN 2017 (Gender) data set:** Given a set of tweets per user (3600 document), the task is to predict the user's gender[4] (Rangel et al., 2017).
- **PAN 2016 (Age) data set:** Given a set of tweets per user (402 documents), the classifier should predict the users's age range[5] (Rangel et al., 2016).
- **MBTI (Meyers-Briggs personality type) data set:** Given a set of tweets per user (8676 documents), the task is to predict to which personality class a user belongs[6], first discussed in Myers (1962).
- **Drug side effects:** This data set links user opinions to side effects of a drug they are taking as treatment. The goal is to predict the side effects prior to experimental measurement (Grässer et al., 2018).[7]
- **Drug effectiveness:** Similarly to side effects (previous data set), the goal of this task is to predict drug effectiveness (Grässer et al., 2018).

Additional statistics for the datasets used in experiments reported in Section 4.2 are provided in Table 1.

Table 1: Data sets used for experimental evaluation of the tax2vec's approach (Section 4.2). Note that MNS corresponds to the maximum number of text segments (max. number of tweets or comments per user or number of news paragraphs as presented in Appendix B).

Data set (target)	Classes	Words	Unique words	Documents	MNS	Average tokens per segment
PAN 2017 (Gender)	2	5169966	607474	3600	102	14.23
MBTI (Personality)	16	11832937	372811	8676	89	27.98
PAN 2016 (Age)	5	943880	178450	402	202	13.17
BBC news	5	902036	58128	2225	76	70.39
Drugs (Side effects)	4	385746	27257	3107	3	41.47
Drugs (Overall effect)	4	385746	27257	3107	3	41.47

2.5 Public datasets for language variety identification

The experiments described in Section 4.3 were conducted on three corpora:

- **DSLCC v4.0** (Tan, Zampieri, Ljubešic, & Tiedemann, 2014)[8]: the corpus used in the VarDial 2017 DSL shared task. The corpus contains 294,000 short excerpts of news texts divided into six distinct language groups (Slavic, Indonesian and Malay, Portuguese, Spanish, French and Farsi) and covering fourteen language varieties in total: Bosnian, Croatian and Serbian; Malay and Indonesian; Persian and Dari; Canadian and Hexagonal French; Brazilian and European Portuguese; Argentine, Peninsular and Peruvian Spanish. Each language contains 20,000 documents for training (out of which 2,000 are to be used as a validation set) and 1,000 for testing.
- **ADIC** (Ali et al., 2015)[9]: the corpus used in the VarDial 2016 ADI shared task. It contains transcribed speech in Modern Standard Arabic, Egyptian, Gulf, Levantine and North African dialects. Speech excerpts were taken from a multi-dialectical corpus containing broadcast, debate and discussion programs from Al Jazeera. Altogether 7,619 documents were used for training (out of which 10% were used for validation) and 1,540 documents for testing.
- **GDIC** (Samardzic et al., 2016): the corpus used in the VarDial 2018 GDI shared task. Texts were extracted from the ArchiMob corpus of Spoken Swiss German[10], which contains 34 oral interviews

[4] https://pan.webis.de/clef17/pan17-web
[5] https://pan.webis.de/clef18/pan18-web
[6] https://www.kaggle.com/datasnaek/mbti-type/kernels
[7] http://archive.ics.uci.edu/ml/datasets
[8] The corpus is publicly available at http://ttg.uni-saarland.de/resources/DSLCC/
[9] The corpus is publicly available at http://alt.qcri.org/resources/ArabicDialectIDCorpus/varDial_DSL_shared_task_2016_subtask2/
[10] The ArchiMob corpus is publicly available at https://www.spur.uzh.ch/en/departments/research/textgroup/ArchiMob.html

with people speaking Bern, Basel, Lucerne and Zurich Swiss German dialects. 19,304 documents were used for training (out of which 10% were used for validation) and 4,752 for testing.

3 Monolingual document linking on Croatian articles

A common feature of online news are references (i.e. links) to other relevant news articles that provide more context or relevant background information. This makes other content relevant for the current story more accessible to readers, while media houses benefit from more efficient use of existing content and improved business metrics such as user engagement and the total time spent on a site. The work described in this section was performed on the data of 24sata[11], described in Section 2.1.

3.1 Experiments with the Croatian dataset of linked articles

Document linking tasks in monolingual scenarios are typically solved by transforming the documents into vectors, which is followed by similarity computation and querying. Vectors can be obtained with many different methods of which TF-IDF models, latent semantic indexing (LSI), and embeddings on the word, sentence[12] or document level are the most frequently used. In our experiments with the Croatian news datasets we experimented with TF-IDF modelling, LSI models of various sizes, and different embeddings.

The dataset of linked news articles contributed by STY contains a list of 24sata articles referencing other articles on the same site. These article references were created by journalists in order to embed links to other related articles in the news article content.

3.1.1 Data preparation and experimental setup

Beside the news content, the article data contains several attributes such as tags, sections, author names etc. which might provide relevant additional information and improve document retrieval. For example, focusing the search to a subset of articles from the same section or to articles containing matching tags should improve results if such information is available during the model inference. Another requirement present in the real system is to take into account the age of the article and preferably return newer articles. Our goal in this section is to compare the methods working with only text so we discard all the supporting metadata.

The text of an article is spread between title, lead and content fields and the first step was to concatenate these three fields. We used three different preprocessing settings, suitable for three data representations used by document matching algorithms: bag-of-words representation, paragraph embeddings, and contextual embeddings.

1. Bag-of-words text representation usually benefits from substantial preprocessing which removes noise and performs normalization. Following tokenization based on regular expression that preserves alphanumeric characters, we filtered out numbers and single character tokens, performed lemmatization with the updated Lemmagen lemmatizer[13] (Jursic et al., 2010), and filtered stopwords using a list of 325 Croatian stopwords.
2. The paragraphs which serve as input to the Doc2Vec model are tokenized with a regular expression that preserves only alphanumeric characters and subsequently lemmatized.

[11] `http://www.24sata.hr`

[12] Sentence/segment level embeddings such as BERT are not very well suited for modelling whole documents using averaging and similarity queries using cosine similarity.

[13] https://github.com/vpodpecan/lemmagen3

3. While the input to contextual embedding models (mBERT and XLM-R) is sometimes slightly preprocessed (e.g., removing the URLs), in our case we performed no preprocesing and used tokenizers provided with the implementation of these models.

The evaluation data consists of 25% of the latest articles from the whole dataset. The reason to choose the latest articles for evaluation is twofold. First, this makes the task of document linking harder because all older articles are potential candidates. Second, this is a more realistic scenario when finding links for a newly published article because all older relevant articles in the database have to be considered. When considering eligible articles in our document retrieval task, we considered their age stored in the *published_date* attribute. This is consistent with the real world scenario where a journalist must not link older but unpublished articles in order to avoid dead links.

The algorithms were trained on the older 75% of articles and evaluated on the latest 25%. The TF-IDF model thus discarded any newly introduced tokens (words) and used IDF estimates from the training data when computing TF-IDF vectors. The same TF-IDF model was used for the LSI model. All embeddings-based models used trained models to infer vectors of the training data. We used cosine similarity for all document retrieval operations[14].

The performance of all algorithms was assessed using the mean average precision score on top ten results returned by the algorithm (MAP@10). This score is calculated by taking the average precision over all results for a single query and calculating the mean value over all those average precisions. The news staff using the implementation of news linking will not be able to browse through all of the results. We believe they can check top results and that correct results that come later will be ignored. For this reason we limit the number of results to ten, such that our metric (MAP@10) is closer to realistic use-case.

TF-IDF

The preprocessing returns a list of tokens for every document. These lists are transformed into sparse numeric vectors by first extracting the corpus vocabulary, computing word frequencies for each document (TF), and computing TF-IDF weighted vectors using the vocabulary and overall word counts.

When compiling the corpus vocabulary additional filtering parameters can be set. We used the common default settings where tokens which appear in less than 5 documents or in more than 50% of all documents are filtered out. In addition, we experimented with setting these limits to 2 and 25%, respectively. The effects of different settings on the performance are presented in Section 3.1.2

In our implementation, we used the `TfidfModel` from the Gensim library (Řehůřek & Sojka, 2010) with the default 'nfc' SMART setting[15] for TF-IDF weighting and normalization which used raw frequency for term frequency weighting, inverse document frequency for document frequency weighting and cosine document length normalization.

LSI

Latent semantic indexing (Deerwester et al., 1990) performs singular value decomposition (SVD) on the weighted term-document matrix which is typically composed of BOW or TF-IDF vectors. The computed SVD is truncated which has the effect of retaining only the most important semantic information while the noise and other artefacts are reduced. The result of LSI is a dense matrix where each document is represented with a fixed dimensional numeric vector (with a few hundred dimensions). In this respect, LSI is similar to embedding methods although the elements of the resulting vectors have a very different meaning.

We used the LSI implementation available in Gensim to transform the same TF-IDF vectors as in the TF-IDF representation above. We tested a number of commonly used target dimensions: 100, 300, and 500.

[14] Note that the cosine similarity might not be the best choice for some embeddings models.

[15] The SMART (System for the Mechanical Analysis and Retrieval of Text) Information Retrieval System defines notation for term weighting and normalization where different formulas are allowed for computing term frequency, document frequency and document length normalization.

Doc2Vec

The Doc2Vec models have a number of hyperparameters that can significantly impact the performance of the model. In order to determine those parameters, we used a Bayesian optimisation. Bayesian optimisation is a strategy for optimization of black-box functions that works by placing a prior belief about the function, and updates it with each evaluation. The parameters of the function for the next evaluation are selected based on a predefined criterion that takes into account previous evaluations of the function. We optimized evaluation metric (MAP@10) on the related articles contained in the training set. To guide the search for parameters, we used a Gaussian Process (GP) prior and Expected Improvement (EI) criterion that maximize the evaluation metric. After 120 evaluations, we selected the best performing hyperparameters. The size of the resulting vector was set to 180 dimensions, the context window covered 5 words left and right from the central word, the vocabulary size was set to 36,000 words, and words with with less than 35 occurrences were ignored. The training procedure used negative sampling with 30 negative words and downsampling of words with frequency higher than $3.7e{-4}$. In Section 3.1.2, we report the results obtained using these hyperparameters.

We used the Doc2Vec (PV-DM and PV-CBOW) implementation available in the Gensim library and the Bayesian optimisation from the Scikit-Optimize[16] library.

mBERT

Multilingual BERT (mBERT) is a 12 layer Transformer model (Vaswani et al., 2017) proposed by Devlin et al. (2019). The mBERT was simultaneously trained on Wikipedia pages of 104 languages. All 104 languages for the mBERT model use shared word piece vocabulary without an explicit way to denote different languages. Maximum length of the input sequence for the model is 512 tokens and each token is represented with 768 dimensions. News articles that can be represented with fewer tokens are padded to the maximum length and articles that require more tokens than the maximum length are trimmed. The input to BERT begins with '*[CLS]*' token and ends with a '*[SEP]*' token denoting the end of a sequence. Additionally, models receives an attention mask to avoid performing attention on padding token indices. Running the model produces a context dependant token representations that we use to create two sequence representations. The first way to represent a sequence is to average token vectors, and the second is to take the representation of only the first token, namely the *[CLS]* token, and run it through the final layer of the model. We evaluate both representations.

We used the *bert-base-multilingual-cased* variant of the pre-trained model available in the Transformers (Wolf et al., 2019) library.

XLM-R

The XLM-R (Conneau et al., 2019) is a large multilingual BERT-like model based on RoBERTa (Liu et al., 2019). It uses the sentence piece tokenizer and is trained as the masked language model on the CommonCrawl data in 100 languages, including Croatian. Similarly to the mBERT, all languages share the same vocabulary (but larger one than mBERT) and the model does not need an explicit marker to denote the language of the input. The maximum size of the input is 512 tokens and each token is represented with 1024 dimension. We are padding shorter news articles with the padding token and trim articles that do not fit in the input. All tokenized sequences begin with the '*<s>*' token that denotes a beginning of the sequence and can also be used for the whole sequence classification. The last token of the sequence is '*<\s>*'. Two representations of sequences are created in the same way as with the mBERT model.

We used the *xlm-roberta-large* variant of the pretrained model available in the Transformers library.

3.1.2 Results

The performance of compared document linking methods using MAP@10 is presented in Table 2.

[16] `https://scikit-optimize.github.io`

The best results were achieved with the baseline TF-IDF model. This is somewhat disappointing for the state-of-the-art neural embeddings. However, one must be aware that we evaluated different representations using only the links between the articles selected by the journalists. This does not necessarily mean that actually retrieved articles are not good recommendations, but verifying this hypothesis requires human evaluation.

The TF-IDF representation consistently performs the best across all evaluated hyperparameters of the model, which do not significantly influence the score. Nevertheless, results in Table 2 suggest that including rare words ($m = 2$) improves the performance, while excluding frequent and rare words ($M = 25\%, m > 2$) decreases the performance. While the difference is small it nevertheless suggests that TF-IDF models trained on this domain may benefit from preprocessing settings that do not remove what is typically considered as artefacts or noise. From a journalist's perspective this corresponds to linking articles based on few rare keywords. This may also offer an explanation why LSI does not achieve scores comparable to TF-IDF. Since LSI is designed to retain only the most important semantic information, the extremes which could improve the models in this particular domain are filtered out.

Doc2Vec approaches the performance of TF-IDF but it does not match it. Doc2Vec shows a significant improvement over LSI, and both variants of Doc2Vec achieve similar score. LSI is less successful with reduction of dimensions but is in general still competitive with much larger mBERT and XLM-R embeddings that use 1024 dimensions to represent documents. For both mBERT and XLM-R, using an average of contextual token embeddings shows better results than using the result of only the *[CLS]* token, which is consistent with conclusions of Reimers & Gurevych (2019).

Table 2: The performance of different approaches on the task of news article retrieval

Model	MAP@10
TF-IDF (m=5, M=50%)[17]	0.279
TF-IDF (m=2, M=50%)	**0.281**
TF-IDF (m=2, M=25%)	**0.281**
TF-IDF (m=10, M=50%)	0.277
TF-IDF (m=10, M=25%)	0.277
LSI (d=500)	0.186
LSI (d=300)	0.166
LSI (d=100)	0.124
Doc2Vec (PV-DM)	0.248
Doc2Vec (PV-CBOW)	0.240
mBERT (AVG)	0.130
mBERT (CLS)	0.007
XLM-R (AVG)	0.167
XLM-R (CLS)	0.047

3.1.3 Discussion on the linked articles dataset

We evaluated several document representations used in recommending related news articles. The results show that the TF-IDF representation produces the results that are more consistent with the manual selection of journalists compared to the results of using more sophisticated article representations. We do not yet have a definite explanation for these outcomes but our belief is that journalists use a kind of keyword search to locate potentially related articles. Related articles found using such a search would contain exactly the same words as the query and would bias the evaluation in favour of the TF-IDF method.

It is possible that the dataset contains a significant amount of noise and that significantly better evaluation results cannot be achieved. For example, one possible source of noise would be due to journalists

[17] This is the default setting for filtering extremes from the dictionary in Gensim.

that misuse the related article information in order to increase the view count of their own articles by adding own unrelated articles to the related articles list. In future work, we plan to perform a manual evaluation of the returned related articles on a selected subset.

Although mBERT and XLM-R models achieve state-of-the-art results in many NLP tasks, they did not fare well in this evaluation. One reason for this might be that document representations created by these model are not suitable for comparison with the cosine similarity. Reimers & Gurevych (2019) reach the same conclusion when evaluating BERT representations and we plan to explore the impact of similarity measures on the performance of mBERT and XLM-R models on this task.

3.2 Experiments on manually annotated triplets

In addition to the dataset of linked articles where journalists selected the related articles, we also used the dataset with related article annotations for further evaluating the performance. The dataset of 5,000 *triplets* is created from a subset of the articles used in experiments in previous subsections. A triplet denotes a set of three articles where a human annotator should provide information about their similarity. All articles from this dataset are from the 24sata newspaper.

Media partner TRI has performed first experiments, using several methods:

- Latent Dirichlet Allocation (LDA): a probabilistic model most suited for topic modelling
- Doc2Vec model, an unsupervised algorithm to generate vectors for a paragraph or a whole document
- a multilingual BERT (mBERT) model pretrained on a Wikipedia data of 104 languages
- a multilingual XLM-RoBERTa (XLM-R) model pretrained on 2.5TB of CommonCrawl data

LDA and Doc2Vec models were inferred on 24sata article data provided by STY and pretrained mBERT and XLM-R were used without training or fine-tuning.

A large number of experiments was performed to evaluate the effect of several hyperparameters. For the LDA, the experiments were performed with different numbers of topics (32–400) and vocabulary sizes (8,000–30,000 tokens). Doc2Vec model was evaluated with differences in preprocessing (with and without stopwords removal), on two variants of the algorithm (Distributed Memory and Distributed Bag-of-Words) with different number of topics (32-400) and vocabulary sizes (8000-30000 tokens). Document representations from mBERT and XLM-R were created in two ways. The first one is to represent the document with the special [CLS] token from the output and the second is to use an average of all output tokens.

Preliminary results were obtained by evaluation of model results on manually annotated triplets provided by TRI. A cosine similarity between article representations was used to estimate article similarity. The evaluation score for the method is the accuracy of the method on the triplet dataset. The LDA algorithm provided results with overall low accuracy (57.0–62.3%). With some variation across experiments, Doc2Vec showed promising results in a variety of settings (62.0–66.5%). A multilingual mBERT model showed very low accuracy on evaluation using a whole sequence [CLS] token embedding (51.1%) and better results when the document was represented by averaging all output tokens (61.3%). The largest model, XLM-R provided better results then the mBERT model. For a whole sequence [CLS] embedding the accuracy (54.5%) was below LDA and averaged token representation yielded the best results (66.9%) for this task. The results for mBERT and XLM-R are consistent with the conclusions from the literature on similar tasks, namely that results provided by a cosine similarity on BERT models do not yield satisfying results and averaging of all tokens is usually better than [CLS] token embedding (see Reimers & Gurevych (2019)). In order to leverage the capabilities of a BERT model created in EMBEDDIA, next steps in T4.1 should take this into account and evaluate an alternative similarity measures for comparing BERT embeddings.

4 Monolingual news categorisation

This section presents our approaches to categorisation and our work on language variety classification. While in the previous section we addressed linking the articles in terms of article retrieval given an input document, in this section we present linking articles in terms of grouping them to predefined topical areas, or to different language groups and varieties.

In terms of linking the articles by categories (i.e. news categorisation), the categories vary with each media source, but very common categories are *sports*, *business*, *politics*, etc. The approaches can either use entire documents, or smaller fragments, such as headlines to classify articles into these areas. In the experiments, we have investigated whether semantic enrichment techniques with WordNet taxonomy can help to improve classification accuracy on a variety of datasets including news segments.

We first present the SRNA method (see Section 4.1) focusing on deep learning approach, followed by *tax2vec* in Section 4.2, which also uses semantic background knowledge, but in a novel approach and much larger experimental setting. In SRNA, we perform news categorisation of Reuters newspaper data set with 46 different topics, while in tax2vec the BBC news with five classes (business, entertainment, politics, sport, tech) is one of the setting. Both settings report results also on a range of other datasets and show how integration of background knowledge can help in news categorisation. The methods could be also further investigated for other types of article linking (see e.g., experiments in the previous section), integrate other type of background knowledge (e.g., keywords and named entities from WP2).[18]

Next, motivated by problems that arise in categorisation and linking if the source media outlet or country of a specific news story is unknown or unconfirmed, we present work on classification of the language variety of a given text, more specifically on short excerpts of news text (see Section 4.3). For example, this can help distinguishing from news from different very closely related languages, such as Croatian, Bosnian, and Serbian, which is interesting when developing datasets for analysing viewpoints. However, the method is more general, presenting integration of n-gram approach and neural architecture, which can serve for other text categorisation tasks.

This section is structured as follows. Section 4.1 presents the SRNA approach to integration of background knowledge into hybrid neural models, followed by the tax2vec approach in Section 4.2 where background knowledge is used for enrichment of vectors in TF-IDF setting (tax2vec features from WordNet taxonomies, as well as Doc2Vec features). In Section 4.3 we report on language variety classification experiments, combining bag-of-n-grams and character level CNN.

4.1 Injecting semantic features into hybrid neural models

It is well known that deep neural networks need a large amount of information in order to learn complex representations from text documents, and that state-of-the-art models do not perform well when incomplete information is used as input (Cho et al., 2015). This work addresses an open problem of increasing the robustness of deep neural network-based classifiers in such settings by exploring to what extent the documents can be truncated without affecting the learner's performance. The approach could also be extended to multilingual setting, as there exist aligned WordNets for several languages.

With this goal in mind, we developed the SRNA approach for leveraging knowledge from taxonomies for construction of novel features for use in a custom deep neural network architecture. The corresponding paper by Škrlj et al. (2019) is provided in Appendix D.

In SRNA (Semantics-aware Recurrent Neural Architecture), semantic information in the form of taxonomies (i.e. ontologies with only hierarchical relations) is propositionalised and then used in a recurrent neural network architecture. The proposed SRNA approach was tested on a document classification

[18] At this stage the methods were not yet tested on media partners' dataset, but as WordNet is multilingual, the methods can also be used in other languages as well as for improving cross-lingual news linking tasks, such as topic modelling presented in Section 5.1.

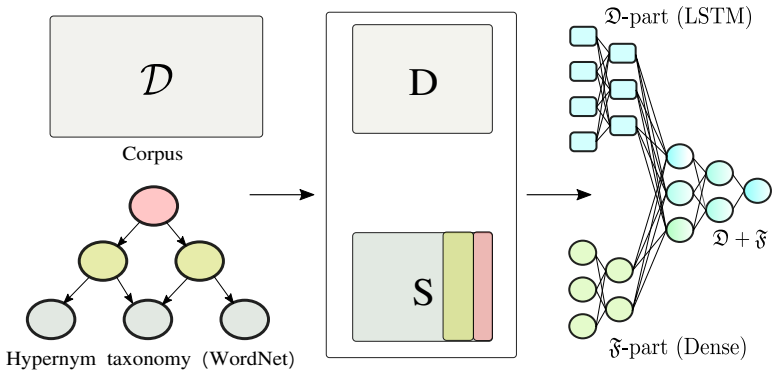


Figure 1: Visualisation of the SRNA approach to semantic space propositionalisation and learning. Left: A document corpus $\mathcal{D}$ and a hypernym taxonomy (WordNet). Middle: A matrix of word indexes D obtained from corpus $\mathcal{D}$, and a matrix of semantic features vectors S (with the same number of rows as D), with features obtained from different levels of the taxonomy. Right: A hybrid neural network architecture is learned from the word index vectors and the semantic feature vectors. Note that sequential word information is present only in the vectors constituting matrix D (word indices), hence part of the architecture exploits sequential information, whereas the constructed semantic features are input to the dense feedforward part of the architecture. Prior to the final layer, intermediary layers of both parts of the network are merged.

task, while special attention was paid to the robustness of the method on short document fragments. Classification of short or incomplete documents is useful in a large variety of tasks. A typical example of short texts are tweets. But, for labelling a news article with a topic tag, using only snippets or titles and not the entire news may be preferred due to limited text availability or required processing speed.

4.1.1 Method description

First, an input corpus $\mathcal{D}$ and a hypernym taxonomy from WordNet are used to construct separate feature matrices D and S. Next, the two matrices are input into a hybrid neural network architecture to predict labels of new input documents.

The second step of the SRNA approach consists of training a deep architecture using the expanded feature matrix (DS) obtained in the first step. In SRNA, semantic features are fed into a deep architecture along with document vectors. The outline of the architecture, shown in Figure 1, can be represented in three main parts. The first part is responsible for learning from document vectors, and is denoted by $\mathfrak{D}$. The second part learns from the constructed semantic vectors, denoted as $\mathfrak{S}$. Finally, before output layer, outputs of $\mathfrak{D}$ and $\mathfrak{S}$ are merged and processed jointly. We denote this part by $(\mathfrak{D}+\mathfrak{S})$.

The recurrent part of the network, represented by the $\mathfrak{D}$ part, is in this work defined as follows. An input vector of word indices is first fed into an embedding layer with dropout regularisation. The resulting output is used in a standard LSTM layer. The output of this step is activated by a ReLU activation function, defined as:

$$ReLU(x) = max(0, x).$$

The output of this layer is followed by a MaxPooling layer. Here, maximal values of a kernel moving across the input vector are extracted. Finally, a dense layer with dropout regularisation is used. Formally, the $\mathfrak{D}$ part of the network can be defined as:

$$\begin{aligned} L_{(1)} &= Dropout(Emb(D)), \\ L_{(2)} &= MaxPooling(ReLU_{(2)}(LSTM(L_{(1)}))), \\ L_{(3w)} &= Dropout(W_{(3)}^{T} L_{(2)} + b_{(3)}). \end{aligned}$$

The $\mathfrak{S}$ part of the architecture similarly consists of fully connected layers. The input for this part of the network are generated semantic features S. It can be represented as:

$$\begin{aligned} L_{(1)} &= Elu_{(1)}(W_{(1)}^T S + b_{(1)}), \\ L_{(2)} &= Dropout(L_{(1)}), \\ L_{(3s)} &= Elu_{(3)}(W_{(3)}^T L_{(2)} + b_{(3)}). \end{aligned}$$

Here, we use the exponential linear unit Clevert et al. (2015), defined as

$$Elu(x) = \begin{cases} x, & \text{for } x \geq 0, \\ c(e^x - 1), & \text{for } x < 0. \end{cases}$$

Here, c is a constant determined during parametrisation of the architecture. Outputs of D and S parts of the architecture are concatenated and used as input to a set of fully connected (dense) layers (M), defined as:

$$\begin{aligned} L_{(1)} &= concat(L_{(3w)}, L_{(3s)}), \\ L_{(2)} &= Elu(Dropout(W_{(2)}^T L_{(1)} + b_{(2)})), \\ L_{(3f)} &= \sigma(W_{(3)}^T L_{(2)} + b_{(3)}). \end{aligned}$$

The *concat* operator merges the outputs of the two individual parts of the network into a single matrix. For concatenation, the dimensions of the two input weight spaces must match (and are reshaped accordingly).

Finally, the output layer $L_{(3f)}$ includes one neuron for each class in the data set. We use binary cross entropy as the loss function. This loss models the outputs as Bernoulli random variables, which offered sufficient performance for the purpose of this paper, whilst also making SRNA suitable for multilabel classification tasks, which are not possible via e.g., softmax-activated outputs.The exact layer parametrisation are discussed in the experimental setting section. The Adam optimiser Kingma & Ba (2014) was chosen due to faster convergence. For the purpose of this deliverable, SRNA was refactored according to the most recent versions of TensorFlow and NLTK libraries, offering (due to the newer TensorFlow static graph engine) even faster training.

4.1.2 Experimental results

We tested the methods on three benchmark data sets, including the Reuters which consists of 11,263 newspaper articles, belonging to 46 different topics (classes). For details see Table 1 in Section 2.

As part of experimental evaluation, we test three deep learning models, two with inclusion of semantic vectors and a baseline ConvNet.

SRNA: Recurrent architecture. This is the proposed architecture that we described in previous subsection. It learns by using LSTM cells on the sequential word indices, and simultaneously captures semantic meaning using dense layers over the semantic feature space.

Baseline RNN. The baseline RNN architecture consists of the non-semantic part of SRNA. Here, a simple unidirectional RNN is trained directly on the input texts.

Baseline CNN. The baseline neural networks used are a 1D convolutional neural network and a recurrent neural network with the same architecture as SRNA, where we omit the semantic part. Here, only word index vectors are used as inputs. The network was parameterised as follows. The number of filters was set to 64, the kernel size used was 5. The MaxPooling region was of size 5. The outputs of the pooling region were used as input to a dense layer with 48 neurons, followed by the final layer.

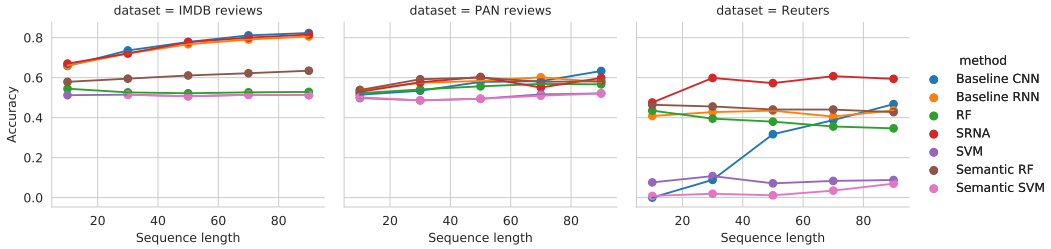


Figure 2: Accuracy results on three benchmark data sets.

As an additional baseline, we implemented also two non-neural classifiers, i.e., the random forest classifier, and a support vector machine, where we also tested how semantic vectors contribute to classification accuracy.

It was observed that, on the Reuters data set (the most relevant for EMBEDDIA), SRNA performed competitively or better in terms of Accuracy and F1, while for non-news data sets it achieved comparable results to baseline RNN and CNN (Figure 2). The poor performance of SVMs could be due to improper scaling and inability to account for the sequential nature of the inputs without prior bag-of-word transformations (they serve as a weak baseline in the paper). For more details, see the paper by Škrlj et al. (2019) attached in Appendix D.

4.2 tax2vec – semantic features from background knowledge

We present a method termed *tax2vec*, which also uses semantic background knowledge, but in a novel approach and much larger experimental setting. The paper by Škrlj et al. (2020) (attached in Appendix E) presents the tax2vec algorithm for semantic feature vector construction that can be used to enrich the feature vectors constructed by the established text processing methods such as TF-IDF. We show that by this enrichment, we manage to improve the performance on a number of classification tasks, including topic classification.

4.2.1 Method description

The tax2vec algorithm takes as input a labelled or unlabelled corpus of n documents and a word taxonomy. It outputs a matrix of *semantic feature vectors* in which each row represents a semantics-based vector representation of one input document. Example use of tax2vec in a common language processing pipeline is shown in Figure 3. Note that the obtained semantic feature vectors serve as additional features in the final, vectorised representation of a given corpus.

Let us first explore how parts of the WordNet taxonomy (Miller, 1995; Fellbaum, 1998) related to the training corpus can be used for the construction of novel features, as such background knowledge can be applied in virtually every English text-based learning setting, as well as for many other languages (Gonzalez-Agirre et al., 2012).

The tax2vec approach implements a two-step semantic feature construction process. First, a document-specific taxonomy is constructed, then a term-weighting scheme is used for feature construction. In the first step of the tax2vec algorithm, a corpus-based taxonomy is constructed from the input document corpus. In this section, we describe how the words from individual documents of a corpus are mapped to terms of the WordNet taxonomy to construct a *document-based taxonomy* by focusing on semantic structures, derived exclusively from the *hypernymy* relation between words. Individual document-based taxonomies are then merged into a joint *corpus-based taxonomy*.

When constructing a document-based taxonomy, each word is mapped to the hypernym WordNet taxonomy. This results in a tree-like structure, which spans from individual words to higher-order semantic

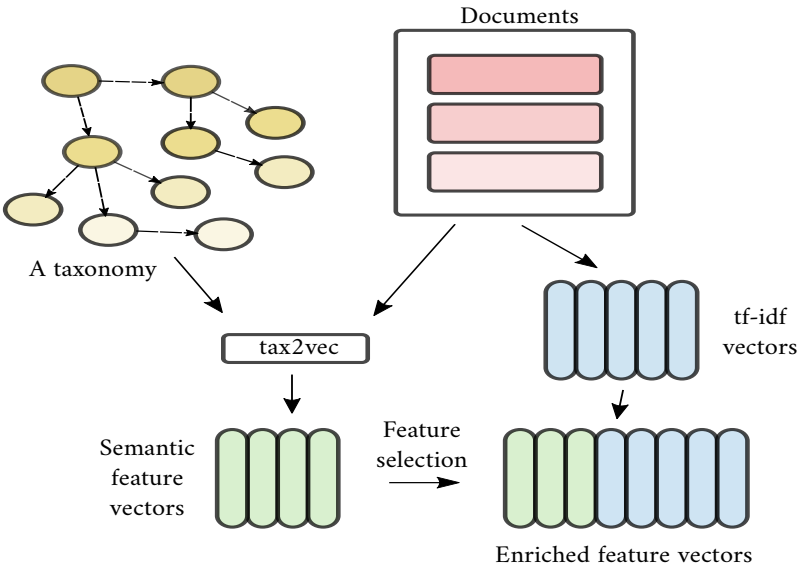


Figure 3: Schematic representation of tax2vec, combined with standard TF-IDF representation of documents. Note that darker nodes in the taxonomy represent more general terms.

$$
\begin{aligned}
&\mathit{Synset}('\mathit{entity}.n.01')\\
\rightarrow\ &\mathit{Synset}('\mathit{abstraction}.n.06')\\
\rightarrow\ &\mathit{Synset}('\mathit{relation}.n.01')\\
\rightarrow\ &\mathit{Synset}('\mathit{part}.n.01')\\
\rightarrow\ &\mathit{Synset}('\mathit{substance}.n.01')\\
\rightarrow\ &\mathit{Synset}('\mathit{chemical\_element}.n.01')\\
\rightarrow\ &\mathit{Synset}('\mathit{astatine}.n.01')
\end{aligned}
$$

Figure 4: Example hypernym path extracted for word “astatine”, where the $\rightarrow$ corresponds to the “hypernym of” relation (the majority of hypernym paths end with the “entity” term, as it represents one of the most general objects in the taxonomy).

concepts. For example, given the word monkey, one of its mappings in the WordNet hypernym taxonomy is the term *mammal*, which can be further mapped to e.g., *animal* etc., eventually reaching the most general term, i.e. *entity*.

In order to construct the mapping, the first problem to be solved is *word-sense disambiguation*. In tax2vec, we use Lesk (Basile et al., 2014), the gold standard WSD algorithm, to map each disambiguated word to the corresponding term in the WordNet taxonomy. The identified term is then associated with a path in the WordNet taxonomy leading from the given term to the root of the taxonomy. Example hypernym path (with WordNet-style notation), extracted for word “astatine”, is shown in Figure 4.

By finding a hypernym path to the root of the taxonomy for all words in the input document, a *document-based taxonomy* is constructed, which consists of all hypernyms of all words in the document. After constructing the document-based taxonomy for all the documents in the corpus, the taxonomies are joined into a *corpus-based taxonomy*.

Note that processing each document and constructing the document-based taxonomy is entirely independent from other documents, allowing us to process the documents in parallel and join the results only when constructing the joint corpus-based taxonomy.

During the construction of a document-based taxonomy, document-level term counts are calculated for each term. For each word t and document D, we count the number $f_{t,D}$ of times the word or one of its

hypernyms appeared in a given document D. The obtained counts can be used for feature construction directly: each term t from the corpus-based taxonomy is associated with a feature, and a document-level term count is used as the feature value. The current implementation of tax2vec weights the feature values using the double normalisation TF-IDF metric. For term t, document D and user-selected normalisation factor K, feature value tf-idf(t,D,K) is calculated as follows:

$$\text{TF-IDF}(t, D, K) = \underbrace{\left(K + (1-K)\frac{f_{t,D}}{\max_{\{t' \in D\}} f_{t',D}}\right)}_{\text{Weighted term frequency}} \cdot \underbrace{\log\left(\frac{N}{n_t}\right)}_{\substack{\text{Inverse} \\ \text{document frequency}}} \tag{1}$$

where $f_{t,D}$ is the term frequency, normalised by $\max_{\{t' \in D\}} f(t', D)$, which corresponds to the raw count of the most common hypernym of words in the document; value N represents the total number of documents in the corpus, n_t denotes the number of document-based taxonomies the hypernym appears in (i.e. the number of documents that contain a hyponym of t). Note that the term frequencies are normalised with respect to the most frequently occurring term to prevent a bias towards longer documents. In the experiments the normalisation constant K was set to 0.5.

The problem with the above presented approach is that all hypernyms from the corpus-based taxonomy are considered, and therefore, the number of columns in the feature matrix can grow to tens of thousands of terms. Including all these terms in the learning process introduces unnecessary noise, and unnecessarily increases the spatial complexity. This leads to the need of feature selection to reduce the number of features to a user-defined number (a free parameter specified as part of the input). We next describe the scoring functions of feature selection approaches considered in this work.

As part of tax2vec, we implemented both supervised (Mutual Information - MI and Personalised PageRank - PPR), as well as unsupervised (Betweenness centrality - BC and term count-based selection) feature selection methods, discussed below. Note that the feature selection process is conducted *exclusively* on the semantic space (i.e. on the mapped WordNet terms).

Feature selection by term counts. Intuitively, the rarest terms are the most document-specific and could provide additional information to the classifier. This is addressed in tax2vec by the simplest heuristic, used in the algorithm: a term-count based heuristic that simply takes overall counts of all hypernyms in the corpus-based taxonomy, sorts them in ascending order according to their frequency of occurrence and takes the top d.

Feature selection using term betweenness centrality. As the constructed corpus-specific taxonomy is not necessarily the same as the WordNet taxonomy, the graph-theoretic properties of individual terms within the corpus-based taxonomy could provide a reasonable estimate of a term's importance. The proposed tax2vec implements the betweenness centrality (BC) (Brandes, 2001) measure of individual terms as the scoring measure. The betweenness centrality is defined as:

$$BC(t) = \sum_{u \neq v \neq t} \frac{\sigma_{uv}(t)}{\sigma_{uv}}; \tag{2}$$

where σ_{uv} corresponds to the number of shortest paths (see Figure 5) between nodes u and v, and $\sigma_{uv}(t)$ corresponds to the number of paths that pass through term (node) t. Intuitively, betweenness measures the t's importance in the corpus-based taxonomy. Here, the terms are sorted in a descending order according to their betweenness centrality, and again, the top d terms are used for learning.

Feature selection using mutual information. The third heuristic, mutual information (MI) Peng et al. (2005), aims to exploit the information from the labels, assigned to the documents used for training. The MI between two random discrete variables represented as vectors F_i and Y (i.e. the i-th hypernym feature and a target binary class) is defined as:

$$MI(F_i, Y) = \sum_{x,y \in \{0,1\}} p(F_i = x, Y = y) \cdot \log_2\left(\frac{p(F_i = x, Y = y)}{p(F_i = x) \cdot p(Y = y)}\right) \tag{3}$$

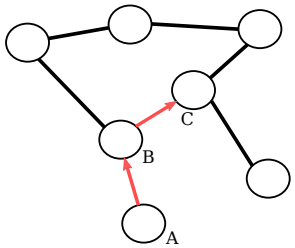


Figure 5: An example shortest path. The path coloured red represents the smallest number of edges needed to reach node C from node A.

where $p(F_i = x)$ and $p(Y = y)$ correspond to marginal distributions of the joint probability distribution of F_i and Y. Note that for this step, tax2vec uses the binary feature representation, where the TF-IDF features are rounded to the closest integer value (either 0 or 1). This way, only well represented features are taken into account. Further, tax2vec uses one-hot encoding of target classes, meaning that each target class vector consists exclusively of zeros and ones. For *each* of the target classes, tax2vec computes the mutual information (MI) between *all* hypernym features (i.e. matrix X) and a given class. Hence, for each target class, a vector of mutual information scores is obtained, corresponding to MI between individual hypernym features and a given target class.

Finally, tax2vec sums the MI scores obtained for each target class to obtain the final vector, which is then sorted in descending order. The first d hypernym features are used for learning. At this point tax2vec yields the selected features as a sparse matrix, maintaining the spatial complexity amounting to the number of float-valued non-zero entries.

Personalised PageRank-based hypernym ranking. Advances by Kralj et al. (2019); Kralj (2017) in learning using extensive background knowledge for rule induction explored the use of Personalised PageRank (PPR) algorithm for node subset selection in semantic search space exploration. In tax2vec, we use the same idea to prioritise (score) hypernyms in the corpus-based taxonomy.

All the aforementioned steps form the basis of tax2vec, outlined in Algorithm 1. First, tax2vec iterates through the given labelled document corpus in parallel (lines 3–7). For each document, *MaptoTaxonomy* method identifies a set of disambiguated words and determines their corresponding terms in taxonomy $\mathfrak{T}$ (i.e. WordNet) using method m (i.e. Lesk). Term counts are stored for later use (*storeTermCounts*), and the taxonomy, derived from a given document (doc) is added to the corpus taxonomy $\mathfrak{T}_{\text{CORPUS}}$. Once traversed, the terms present in $\mathfrak{T}_{\text{CORPUS}}$ represent potential *features*. Term counts, stored for each document are aggregated into n vectors, where n is the number of documents in the corpus. The result of this step is a real-valued, sparse matrix (vecSpace), where columns represent all possible terms from $\mathfrak{T}_{\text{CORPUS}}$. In the following step, feature selection is conducted. Here, graph-based methods (e.g., BC and PPR) identify top d terms based on $\mathfrak{T}_{\text{CORPUS}}$'s properties (lines 9–12), and non-graph methods (e.g., MI) is used directly on the sparse matrix to select which d features are the most relevant (lines 13–15). Finally, *selectedFeatures*, a matrix of selected semantic features is returned.

4.2.2 Experimental results

As tax2vec serves as a preprocessing method for data enrichment with semantic features, arbitrary classifiers can use the resulting semantic features for learning. Note that in the experiments, the final feature space is composed of both semantic and non-semantic (original) features, i.e., the final feature set used for learning is formed *after* the semantic features have been constructed and selected, by concatenating the original features and the semantic features. We use the following learners:

PAN 2017 approach. An SVM-based approach that relies heavily on the method proposed by Martinc et al. (2017) for the author profiling task in the PAN 2017 shared task (Rangel et al., 2017). In contrast

Algorithm 1: tax2vec

Data: Training set documents D, training document labels Y_{tr}, WordNet taxonomy $\mathfrak{T}$, word-to-taxonomy mapping m, feature selection heuristic h, number of selected features d

$\mathfrak{T}_{\text{CORPUS}} \leftarrow$ empty structure;
termCounts $\leftarrow$ empty structure;
for *doc* $\in D$ *(in parallel)* **do**
  $\mathfrak{T}_{\text{DOCUMENT}} \leftarrow$ MaptoTaxonomy($doc, \mathfrak{T}, m$);
  Add storeTermCounts($\mathfrak{T}_{\text{DOCUMENT}}$) to termCounts;
  Add $\mathfrak{T}_{\text{DOCUMENT}}$ to $\mathfrak{T}_{\text{CORPUS}}$;
end
vecSpace $\leftarrow$ TF-IDF(constructTfVectors($D, \mathfrak{T}_{\text{CORPUS}}$,termCounts));
if *h is graph-based* **then**
  topTerms $\leftarrow$ selectFeatures(h, $\mathfrak{T}_{\text{CORPUS}}$, d, optional Y_{tr});
  selectedFeatures $\leftarrow$ select topTerms from vecSpace;
end
else
  selectedFeatures$\leftarrow$ selectFeaturesDirectly(h, vecSpace,d ,Y_{tr});
end
return selectedFeatures;

Result: d new feature vectors in sparse vector format.

to the original approach, we do not use POS tag sequences as features and a Logistic regression classifier is replaced by a Linear SVM. Here, we experimented with the regularisation parameter C, for which values in range $\{1, 20, 50, 100, 200\}$ were tested. This SVM variant is from this point on referred to as "SVM (Martinc et al.)". As this feature construction pipeline consists of too many parameters, we were not able to perform extensive grid search due to computational complexity. Thus, we did not experiment with feature construction parameters, and kept the configuration proposed in the original study.

Linear SVM with automatic feature construction. The second learner is a libSVM linear classifier (Chang & Lin, 2011), trained on a predefined number of word and character level n-grams, constructed using Scikit-learn's *TfidfVectorizer* method. To find the best setting, we varied the SVM's C parameter in range $\{1, 20, 50, 100, 200\}$, the number of word features between $\{10000, 50000, 100000, 200000\}$ and character features between $\{0, 30\}$. Note that the word features were sorted by decreasing frequency. Here, we considered (word) n-grams of lengths between two and six. This SVM variation is from this point on referred to as "SVM (generic)". The main difference between "SVM (generic)" and "SVM (Martinc et al.)" is that the latter approach also considers punctuation-based and suffix-based features. Further, it is capable of constructing features that represent document sentiment, which was proven to work well for social media data sets (e.g., tweets). Finally, Martinc's approach also accounts for character repetitions and has a parameter for social-media text cleaning in preprocessing. Note that for both SVM approaches we fine-tuned the hyperparameter C, as is common when employing SVMs, and scaled as done in Martinc et al.'s approach. The hyperparameter values govern how penalised the learner is for a mis-classified instance, which is a property that was shown to vary across data sets (see for example Meyer et al. (2003)).

Hierarchical attention networks (HILSTM). The first neural network baseline is the recently introduced hierarchical attention network (Yang et al., 2016). Here, we performed a grid search over $\{64, 128, 256\}$ hidden layers sizes, embedding sizes of $\{128, 256, 512\}$, batch sizes of $\{8, 24, 52\}$ and number of epochs $\{5, 15, 20, 30\}$.

Deep feedforward neural networks. As tax2vec constructs feature vectors, we also attempted to use them as inputs for a standard feedforward neural network architecture (LeCun et al., 2015). Here, we performed a grid search across hidden layer settings: $\{(128, 64), (10, 10, 10)\}$ (where for example $(128, 64)$ corresponds to a two hidden layer neural network, where in the first hidden layer there

are 128 neurons and 64 in the second), batch sizes $\{8, 24, 52\}$ and the number of training epochs $\{5, 15, 20\}$.[19]

In addition to the semantic features constructed by tax2vec, Doc2Vec-based semantic features (Le & Mikolov, 2014) were used as a baseline in order to allow for a simple comparison between two semantic feature construction approaches. They were concatenated with the features constructed by Martinc et al.'s SVM approach, in order to compare the benefits merging the BoW-based representations with a different type of semantic features (embedding-based ones). We set the embedding dimension to 256, as it was shown that lower dimensional embeddings do not perform well (Pennington et al., 2014).

The experiments were set up as follows. For the drug-related data sets, we used the splits given in the original paper Grässer et al. (2018). For other data sets, we trained the classifiers using stratified $90\% : 10\%$ splits. For each classifier, 10 such splits were obtained. The measure used in all cases is F_1, where for the multiclass problems (e.g., MBTI), we use the micro-averaged F_1. All experiments were repeated five times using different random seeds. The features obtained using tax2vec are used in combination with SVM classifiers, while the other classifiers are used as baselines.[20]

The F_1 results are presented in Table 3. The first observation is that combining BoW-based representations with semantic features (tax2vec or Doc2Vec) leads to performance improvements in five out of six cases (MBTI being the only data set where no improvement is detected). Tax2vec outperforms Doc2Vec-based vectors in three out of five data sets (PAN 2016 (Age), BBC News and Drugs (effect)), while Doc2Vec-based features outperform tax2vec on two data sets (PAN 2017 (gender) and Drugs (Side)).

Table 3: Effect of the added semantic features to classification performance, where all text segments (tweets/comments per user or segments per news article) are used. The best performing feature selection heuristic for the majority of top performing classifiers was "rarest terms" or "Closeness centrality", indicating that only a handful of hypernyms carry added value, relevant for classification. Note that the results in the table correspond to the best performing combination of a classifier and a given heuristic.

# Semantic	Learner	PAN (Age)	PAN (Gender)	MBTI	BBC News	Drugs (effect)	Drugs (side)
0	HILSTM	0.422	0.752	0.407	0.833	0.443	0.514
0	SVM (Martinc et al.)	0.417	0.814	0.682	0.983	0.468	0.503
0	SVM (generic)	0.424	0.751	0.556	0.967	0.445	0.462
256 (Doc2Vec)	SVM (Martinc et al.)	0.422	0.817	0.675	0.979	0.416	0.523
30 (tax2vec)	DNN	0.400	0.511	0.182	0.353	0.400	0.321
10 (tax2vec)	SVM (Martinc et al.)	0.445	0.815	0.679	0.996	0.47	0.506
	SVM (generic)	0.502	0.781	0.556	0.972	0.445	0.469
25 (tax2vec)	SVM (Martinc et al.)	0.454	0.814	0.681	0.984	0.468	0.500
	SVM (generic)	0.484	0.755	0.554	0.967	0.449	0.466
50 (tax2vec)	SVM (Martinc et al.)	0.439	0.814	0.681	0.983	0.462	0.499
	SVM (generic)	0.444	0.751	0.554	0.963	0.446	0.463
100 (tax2vec)	SVM (Martinc et al.)	0.424	0.816	0.678	0.984	0.466	0.496
	SVM (generic)	0.422	0.749	0.551	0.958	0.443	0.46
500 (tax2vec)	SVM (Martinc et al.)	0.383	0.797	0.662	0.975	0.45	0.477
	SVM (generic)	0.400	0.724	0.532	0.909	0.424	0.438
1000 (tax2vec)	SVM (Martinc et al.)	0.368	0.783	0.647	0.964	0.436	0.466
	SVM (generic)	0.373	0.701	0.512	0.851	0.407	0.420

When it comes to tax2vec, up to 100 semantic features aid the SVM learners to achieve better accuracy. The most apparent improvement can be observed for the case of PAN 2016 (Age) data set, where the task was to predict age. Here, 10 semantic features notably improved the classifiers' performance (up to approximately 7% for SVM (generic)). Further, a minor improvement over the state-of-the-art was also observed on the PAN 2017 (Gender) data set and the BBC news categorisation (see results for SVM (Martinc et al.)). Hierarchical attention networks outperformed all other learners for the task

[19] The two deep architectures were implemented using TensorFlow (Abadi et al., 2015), and trained using a Nvidia Tesla K40 GPU. We report the best result for top 30 semantic features with the rarest terms heuristic.

[20] Note that simple feedforward neural networks could also be used in combination with hypernym features—we leave such computationally expensive experiments for further work.

Table 4: Most informative features in the BBC News data set with respect to the target class (ranked by MI)—Classes represent news topics). Individual target classes are sorted according to a descending mutual information with respect to a given feature.

		Sorted target class-mutual information pairs			
Semantic feature	Average MI	Class 1	Class 2	Class 3	Class 4
tory.n.03	0.057	politics:0.14	entertainment:0.05	business:0.03	sport:0.01
movie.n.01	0.059	business:0.14	politics:0.04	entertainment:0.04	sport:0.02
conservative.n.01	0.061	politics:0.15	entertainment:0.05	business:0.03	sport:0.01
vote.n.02	0.061	business:0.15	entertainment:0.04	politics:0.04	sport:0.02
election.n.01	0.063	entertainment:0.16	business:0.05	politics:0.04	sport:0.0
topology.n.04	0.063	entertainment:0.16	business:0.05	politics:0.04	sport:0.0
mercantile_establishment.n.01	0.068	politics:0.17	business:0.07	entertainment:0.03	sport:0.01
star_topology.n.01	0.069	politics:0.17	business:0.07	entertainment:0.03	sport:0.01
rightist.n.01	0.074	politics:0.18	business:0.06	entertainment:0.04	sport:0.01
marketplace.n.02	0.087	entertainment:0.22	business:0.06	politics:0.05	sport:0.01

of side effects prediction, yet semantics-augmented SVMs outperformed neural models when general drug effects were considered as target classes. Similarly, no performance improvements were offered by tax2vec on the MBTI data set.

As discussed in the previous sections, tax2vec selects a set of hypernyms according to a given heuristic and uses them for learning. One of the key benefits of such approach is that the selected semantic features can easily be inspected, hence potentially offering interesting insights into the semantics, underlying the problem at hand. We discuss here a set of 30 features which emerged as relevant according to the "mutual information" heuristic when the BBC News data set was considered. Here, tax2vec was trained on 90% of the data, the rest was removed (test set). The features and their corresponding mutual information scores are shown in Table 4.

We can observe that the "sport" topic (BBC data set) is not well associated with the prioritised features. On the contrary, terms such as "rightist" and "conservative" emerged as relevant for classifying into the "politics" class. Similarly, "marketplace" for example, appeared relevant for classifying into the "entertainment" class.

We repeated a similar experiment using the "rarest terms" heuristic. The terms which emerged are:

> 'problem.n.02', 'question.n.02', 'riddle.n.01', 'salmon.n.04', 'militia.n.02', 'orphan.n.04', 'taboo.n.01', 'desertion.n.01', 'dearth.n.02', 'outfitter.n.02', 'scarcity.n.01', 'vasodilator.n.01', 'dilator.n.02', 'fluoxetine.n.01', 'high blood pressure.n.01', 'amlodipine besylate.n.01', 'drain.n.01', 'imperative mood.n.01', 'fluorescent.n.01', 'veneer.n.01', 'autograph.n.01', 'oak.n.02', 'layout.n.01', 'wall.n.01', 'firewall.n.03', 'workload.n.01', 'manuscript.n.02', 'cake.n.01', 'partition.n.01', 'plasterboard.n.01'

Even if the feature selection method is unsupervised (not directly associated to classes), we can immediately observe that the features correspond to different topics, ranging from medicine (e.g., "high blood pressure"), politics (e.g., "militia") to food(e.g., "cake") and more, indicating that the rarest hypernyms are indeed diverse and as such potentially useful for the learner.

The results suggest that tax2vec could potentially also be used to inspect the semantic background of a given data set directly, regardless of the learning task.

4.2.3 Extensions and further work

The current version of tax2vec is one of the first approaches that explored how unsupervised feature ranking can aid in selection of potentially useful semantic space for a given down-stream learning task. However, multiple aspects could be further developed, and are discussed next. First, tax2vec focuses on the english domain, albeit taxonomies can span across languages or are available for a different

language entirely. As such, tax2vec shall be extended to perform in cross-lingual setting by exploiting multilingual taxonomies, into which tokens from a given language can be mapped. As such, tax2vec could be applied to texts in arbitrary languages, extending its functionality significantly. Further, the cross-lingual embeddings could be used alongside the remainder of the feature space, potentially improving the performance, as the inclusion of BoW, semantic and latent features could capture various aspects of a given document, from character level morphological features to semantic context. As such, tax2vec could aid in development of approaches suitable for *low resource learning*.

4.3 Language variety classification

Task 4.1 aims to develop news linking methods capable of linking news stories from different languages and media sources. The problems arises if the source media outlet or country of a specific news story is unknown or unconfirmed, which is not uncommon due to recent rise of fake news and misinformation (Lazer et al., 2018). This phenomenon makes the analysis of the differences in news reporting in different countries unreliable ans is especially detrimental for Embeddia languages that are spoken in more than one country, since the reporting in different political and cultural entities can differ significantly. We tackled the problem of differentiating between similar language varieties and similar languages in the study by Martinc & Pollak (2019), presented below and included in full in Appendix B.

4.3.1 Language variety classifier architecture

The related work on language variety classification (Belinkov & Glass, 2016; Bjerva, 2016) indicates that using character-level CNNs might be the most promising neural approach to the task of discriminating between similar languages. CNNs are able to identify important parts of a text sequence by employing a max-over-time pooling operation (Collobert et al., 2011), which keeps only the character sequences with the highest predictive power in the text. These sequences of predefined lengths resemble character n-grams, which were used in nearly every winning approach in the past language variety shared task (Zampieri et al., 2017; Malmasi et al., 2016; Rangel et al., 2017), but the CNN approach also has the advantage over the traditional bag-of-n-grams (BON) approaches, that it preserves the order in which these text areas with high predictive power appear in the text.

On the other hand, its main disadvantage could be the lack of an effective weighting scheme that would be capable of determining how specific these character sequences are for every input document. The data is fed into a neural classifier in small batches, therefore it is impossible for it to obtain a somewhat global view on the data and its structure, which is encoded in the more traditional TF-IDF (or BM25 (Robertson & Zaragoza, 2009)) weighted input matrix. Another intuition that might explain the usefulness of weighting schemes for the specific task of language variety classification is related to named entities, for which it was shown in the past shared tasks that they in many cases reflect the origin of the text (Zampieri et al., 2015). The hypothesis is that these entities are quite rare and somewhat document specific and are therefore given large weights by different weighting schemes, encouraging the classifier to pay attention to them. The importance of choosing an effective weighting scheme on the task of discriminating between similar languages is also emphasised in the research by Bestgen (2017), the winner of the VarDial 2017 DSL task, who managed to gain some performance boost by replacing the TF-IDF weighting scheme with BM25.

Our architecture (visualised in Figure 6) builds on these findings from the literature and is in its essence an effective hybrid between a traditional feature engineering approach, which relies on different kinds of BON features, and a newer neural feature engineering approach to text classification. This combination of two distinct text classification architectures is capable of leveraging character-level and more global document/corpus-level information and achieving synergy between these two data flows. The main idea is to improve on standard CNN approaches by adding an additional input to the network that would overcome the lack of an effective weighting scheme. Therefore, the text is fed to the network in the form of two distinct inputs (as presented in Figure 6):

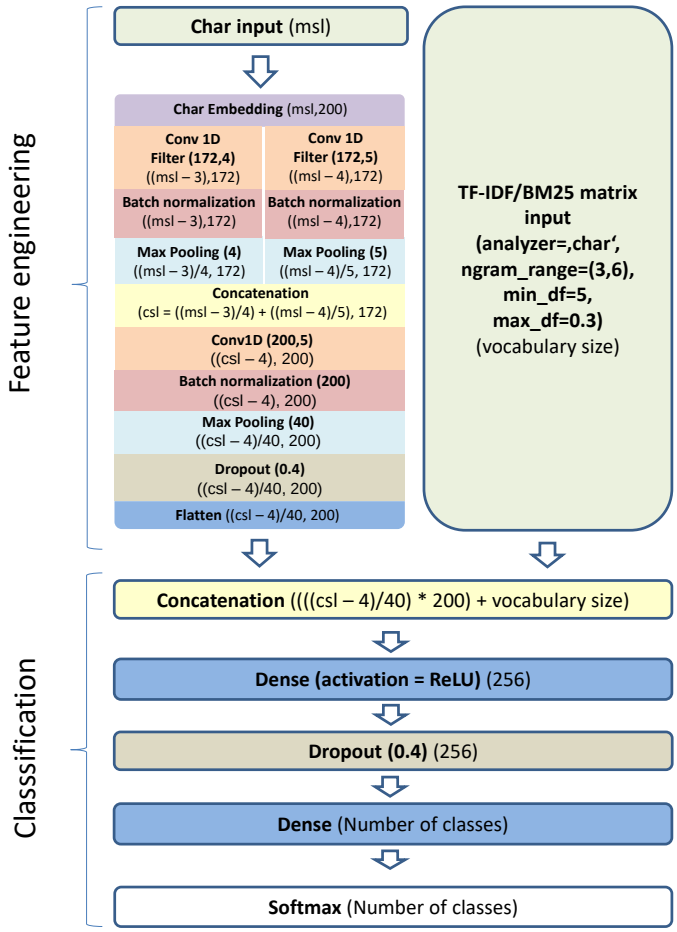


Figure 6: System architecture: layer names and input parameters are written in bold, layer output sizes are written in normal text, *msl* stands for maximum sequence length and *csl* stands for concatenated sequence length.

- *Char input*: Every document is converted into a numeric character sequence (every character is represented by a distinct integer) of length corresponding to the number of characters in the longest document in the train set (zero value padding is added after the document character sequence and truncating is also performed at the end of the sequence if the document in the validation or test set is too long).
- *TF-IDF/BM25 matrix*: We explore the effect of two distinct weighting schemes on the performance of the classifier, therefore input dataset is converted into a matrix of either TF-IDF or BM25 weighted features with a *TfidfVectorizer* from ScikitLearn (Pedregosa et al., 2011) or our own implementation of the *BM25Vectorizer*. The matrix is calculated on character n-grams of sizes three, four, five and six with a minimum document frequency of five and appearing in at most thirty percent of the documents in the train set. Sublinear term frequency scaling is applied in the term frequency calculation when *TfidfVectorizer* is used and for BM25 weighting parameters b and k_1 are set to 0.75 and 1.2 respectively, same as in Bestgen (2017).

The architecture for processing *Char input* is a relatively shallow character-level CNN with randomly initialised embeddings of size $msl \times 200$, where *msl* stands for *maximum sequence length*. Assuming that w is a convolutional filter, b is a bias and f a non-linear function (a rectified linear unit (*ReLU*) in our case), a distinct character n-gram feature c_i is produced for every possible window of h characters $x_{i:i+h-1}$ in the document as follows:

$$c_i = f(w \cdot x_{i:i+h-1} + b)$$

In the first step, we employ two parallel convolutional layers (one having a window of size four and the other of size five), each of them having 172 convolutional filters. These layers return two feature maps of size $(msl - ws + 1) \times 172$, where ws is the window size. Batch normalisation and max-over-time pooling operations are applied on both feature maps in order to filter out features with low predictive power. These operations produce two matrices of size $(msl - ws + 1)/mws \times 172$, where sizes of max-pooling windows (mws) correspond to convolution window sizes. Output matrices are concatenated and the resulting matrix is fed into a second convolutional layer with 200 convolutional filters and window size five. Batch normalisation and max-over-time pooling are applied again and after that, we conduct a dropout operation on the output of the layer, in which forty percent of input units are dropped in order to reduce overfitting. Finally, the resulting output is flattened (changed from a two-dimensional to a one dimensional vector) and passed to a *Concatenation* layer, where it is concatenated with the input *TF-IDF/BM25 matrix*. The resulting concatenation is passed on to a fully connected layer (*Dense*) with a *ReLU* activation layer and dropout is conducted again, this time on the concatenated vectors. A final step is passing the resulting vectors to a dense layer with a *Softmax* activation, responsible for producing the final probability distribution over language variety classes.

4.3.2 Experiments

We tested the proposed approach on the **DSLCC v4.0** (Tan, Zampieri, Ljubešic, & Tiedemann, 2014)[21] corpus used in the VarDial 2017 DSL shared task (Zampieri et al., 2017) (Corpus statistics are presented in Table 5). The corpus contains 294 000 short excerpts of news texts divided into six distinct language groups (Slavic, Indonesian and Malay, Portuguese, Spanish, French and Farsi) and covering fourteen language varieties in total: Bosnian, Croatian and Serbian; Malay and Indonesian; Persian and Dari; Canadian and Hexagonal French; Brazilian and European Portuguese; Argentine, Peninsular and Peruvian Spanish. Each language contains 20,000 documents for training (out of which 2000 are to be used as a validation set) and 1000 for testing.

Table 5: DSLCC v4.0 corpus

DSLCC v4.0					
Language/Variety	Class	Train inst.	Train tokens	Test inst.	Test tokens
Bosnian	bs	20 000	716 537	1 000	35 756
Croatian	hr	20 000	845 639	1 000	42 774
Serbian	sr	20 000	777 363	1 000	39 003
Indonesian	id	20 000	800 639	1 000	39 954
Malay	my	20 000	591 246	1 000	29 028
Brazilian Portuguese	pt-BR	20 000	907 657	1 000	45 715
European Portuguese	pt-PT	20 000	832 664	1 000	41 689
Argentine Spanish	es-AR	20 000	939 425	1 000	42 392
Castilian Spanish	es-ES	20 000	1 000 235	1 000	50 134
Peruvian Spanish	es-PE	20 000	569 587	1 000	28 097
Canadian French	fr-CA	20 000	712 467	1 000	36 121
Hexagonal French	fr-FR	20 000	871 026	1 000	44 076
Persian	fa-IR	20 000	824 640	1 000	41 900
Dari	fa-AF	20 000	601 025	1 000	30 121
Total		280 000	8 639 459	14 000	546 790

We chose to use a two-step approach, as first proposed by Goutte et al. (2014):

[21] The corpus is publicly available at `http://ttg.uni-saarland.de/resources/DSLCC/`

1. The general classifier is trained to identify the language group for every specific document. For this step, the input TF-IDF/BM25 matrix is calculated only on the word bound character n-grams[22] of sizes three, four and five with a minimum document frequency of five and appearing in at most thirty percent of the documents in the train set. This configuration produces a TF-IDF/BM25 matrix of smaller size than if the configuration for the TF-IDF/BM25 matrix, described in Section 4.3.1, was used. This size reduction was chosen because distinguishing between different language groups is not a difficult problem, therefore this parameter reduction does not influence performance but it reduces the execution time.
2. We train six different classification models, one for each language group. After being classified as belonging to a specific language group by the general classifier in Step 1, the documents are assigned to the appropriate classifier for predicting the final language variety.

Since NLP tools and resources such as part-of-speech taggers, pretrained word embeddings, word dictionaries and tokenizers might not exist for some under-resourced languages, we also believe that an architecture which does not require language specific resources and tools, apart from the training corpus, might be more useful and easier to use in real-life applications. For this reason, our system does not require any additional resources and the conducted preprocessing procedure is light[23].

We show (see Table 6) that the proposed architecture is generic enough to outperform the winning approach of VarDial 2017 on all of the language groups without any language group specific parameter or architecture tweaking. In contrast, most of the approaches of the VarDial 2017 DSL shared task resorted to language-group specific optimisation, as getting even the slightest possible performance boost by employing this tactic was important due to the competitive nature of shared tasks.

We conducted an extensive grid search on the DSLCC v4.0 in order to find the best hyperparameters for the model. All combinations of the following hyperparameter values were tested before choosing the best combination, which is written in bold in the list below and presented in Section 4.3.1:

- Learning rates: 0.001, **0.0008**, 0.0006, 0.0004, 0.0002
- Number of parallel convolutions with different filter sizes: [3] [4], [3,4], **[4,5]**, [5,6], [6,7], [3,4,5], [4,5,6], [5,6,7], [3,4,5,6], [4,5,6,7], [3,4,5,6,7]
- Character embedding sizes: 100, **200**, 400
- Dense layer sizes: 128, **256**, 512
- Dropout values: 0.2, 0.3, **0.4**, 0.5
- Number of convolutional filters in the first convolution step: 156, **172**, 200
- Number of convolutional filters in the second convolution step: 156, 172, **200**
- Size of a max-pooling window in the second convolution step: 10, 20, **40**, 60
- BON n sizes: [3] [4] [3,4], [4,5], [5,6], [6,7], [3,4,5], [4,5,6], [5,6,7], **[3,4,5,6]**, [4,5,6,7], [3,4,5,6,7]
- Minimum document frequency of an n-gram in the TF-IDF/BM25 matrix: [2], **[5]**, [10]
- BM25 b parameter: 0.5, **0.75**, 1.0
- BM25 k_1 parameter: 1.0, **1.2**, 1.4

The hyperparameters, which influenced the performance of the network the most, were the learning rate, CNN filter sizes, size of the max-pooling window, BON n size and a minimum document frequency of n-grams. Too many parallel convolutions, small sizes of the max-pooling window and low minimum document frequency of n-grams showed tendency towards overfitting, especially when used together

[22] Word bound character n-grams are made only from text inside word boundaries, e.g., a sequence *this is great* would produce a word bound character 4-gram sequence *this, is__, grea, reat*, in which _ stands for empty space character.

[23] We only replace all email addresses in the text with *EMAIL* tokens and all URLs with *HTTPURL* tokens by employing regular expressions. Even if this might not be relevant to all of the corpora, we keep the preprocessing unchanged for all the settings.

in combination. In general, we noticed quite a strong tendency towards overfitting no matter the hyper-parameter combination, which could be to some extent the consequence of feeding a high dimensional TF-IDF/BM25 matrix to the network, which greatly increases the number of network parameters. We noticed that a combination of a relatively small learning rate and a large dropout worked best to counter this tendency.

Another thing we noticed is that using exactly the same configurations of convolutional filter sizes and n-gram sizes negatively affected the performance, which was slightly improved when the configurations did not completely overlap. The hypothesis is that synergy between two data flows is less effective if the information in these two data flows is too similar. The validation set results did however show that configurations containing 4- and 5-grams and filter sizes of 4 and 5 in general worked better than other configurations for DSLCC v4.0 classification, therefore these configurations were used in both data flows despite the overlap.

We use the Python Keras library (Chollet et al., 2015) for the implementation of the system. For optimisation, we use an Adam optimiser (Kingma & Ba, 2014) with a learning rate of 0.0008. For each language variety in the DSLCC v4.0, the model is trained on the train set for twenty epochs and tested on the validation set after every epoch.

Table 6 presents the results achieved by our neural classifier on the DSLCC v4.0 corpus in comparison to the winner of the VarDial 2017 DSL shared task (Bestgen, 2017) in terms of weighted F1, micro F1, macro F1 and accuracy measures. The first step of the two-step classification approach, distinguishing between different language groups (*All-language groups (TF-IDF)* and *All-language groups (BM25)* rows in Table 6), proved trivial for the system, which achieved almost perfect weighted F1 score and misclassified only twenty-seven documents out of 14 000 in the test set when TF-IDF weighting scheme was used and twenty-nine documents when BM25 weighting scheme was used.

Table 6: Results of the proposed language variety classifier on the DSLCC v4.0 for different language groups, as well as for the discrimination between language groups (All-language groups). Also the results for all language varieties (All-language varieties) are provided, for which a comparison with the official VarDial 2017 winners is made. Results for both weighting schemes, TF-IDF and BM25, are reported separately.

Language group (weighting)	F1 (weighted)	F1 (micro)	F1 (macro)	Accuracy
All-language groups (TF-IDF)	**0.9981**	**0.9981**	**0.9980**	**0.9981**
All-language groups (BM25)	0.9979	0.9979	0.9980	0.9980
Spanish (TF-IDF)	**0.9136**	**0.9140**	**0.9136**	**0.9140**
Spanish (BM25)	0.9042	0.9047	0.9042	0.9047
Slavic (TF-IDF)	0.8645	0.8650	0.8645	0.8650
Slavic (BM25)	**0.8752**	**0.8753**	**0.8752**	**0.8753**
Farsi (TF-IDF)	0.9685	0.9685	0.9685	0.9685
Farsi (BM25)	**0.9690**	**0.9690**	**0.9690**	**0.9690**
French (TF-IDF)	**0.9570**	**0.9570**	**0.9570**	**0.9570**
French (BM25)	0.9545	0.9545	0.9545	0.9545
Malay and Indonesian (TF-IDF)	0.9855	0.9855	0.9855	0.9855
Malay and Indonesian (BM25)	**0.9860**	**0.9860**	**0.9860**	**0.9860**
Portuguese (TF-IDF)	**0.9480**	**0.9480**	**0.9480**	**0.9480**
Portuguese (BM25)	0.9460	0.9460	0.9460	0.9460
All-language varieties (TF-IDF)	**0.9310**	**0.9312**	**0.9310**	**0.9312**
All-language varieties (BM25)	0.9304	0.9305	0.9304	0.9305
VarDial 2017 winner (Bestgen, 2017)	0.9271	0.9274	0.9271	0.9274

The results for the second step of the two-step classification approach indicate that the difficulty of distinguishing language varieties within different language groups varies. The system had most difficulties with distinguishing between different Slavic languages, where it achieved by far the worst results with an weighted F1 of 0.8645 when TF-IDF weighting scheme was employed and about one percentage point better results when BM25 weighting was used. The second most difficult were Spanish variet-

ies. We should point out that this comes as no surprise, since Slavic and Spanish languages groups were the only two groups that contained three varieties, while the other groups in DSLCC v4.0 contained two varieties. The system had least problems with distinguishing between Malay and Indonesian languages.

When it comes to comparing two weighting schemes, there is no clear overall winner. The biggest differences in performance are on Spanish varieties, where TF-IDF weighting outperforms BM25 by about one percentage point according to every measure, and on Slavic varieties, where BM25 weighting outperforms TF-IDF by a very similar margin. The differences on other varieties are smaller, ranging from 0.005 on Farsi and Malay and Indonesian varieties to 0.020 on Portuguese varieties.

4.3.3 Error analysis

We conducted a manual error analysis on the misclassified Slavic documents[24] in order to get a clearer picture about what kind of documents are the hardest to classify. Misclassified documents were manually grouped into four classes according to the number and type of named entities found in the document:

- **No named entities**: Documents without any named entities
- **Misleading named entities**: Documents containing any named entities (e.g., names of regions, cities, public figures...) originating from a country with the official language variety corresponding to one of the two possible incorrect language varieties (e.g., a document labelled as Serbian containing the word *Zagreb*, which is the capital of Croatia, would be put into this class).
- **Clarifying named entities**: Documents containing named entities originating from a country with the official language variety being the correct language variety and containing no misleading entities.
- **Unrelated named entities**: Documents containing only named entities that are not originating from any of the countries speaking target language varieties (e.g., a document containing only the named entity *Budapest* would be classified into this category).

Results of the analysis are presented in Table 7. The results show that a large portion of misclassified documents (73%) either contain no named entities (36%) or contain only unrelated named entities (37%), which might make them harder to classify, although we can not claim that for sure, since we do not know the distribution of these classes across the entire test set. 17% of the documents on the other hand contain misleading named entities that could influence the classifier prediction. There are also 41 documents (10%) containing only clarifying named entities that would be easily classified correctly by any human annotator with some basic background knowledge about Serbia, Bosnia and Croatia. This suggests that there is still some room for improvement for the developed classifier.

Table 7: Results of the error analysis on 405 misclassified Slavic documents.

Group	Num. doc.	Prop. of doc.	Avg. doc. length
No named entities	144	0.36	26.94
Misleading named entities	70	0.17	40.96
Clarifying named entities	41	0.10	34.96
Unrelated named entities	150	0.37	33.17
All misclassified	405	1.00	32.48

Another finding is that misclassified documents are in average shorter (32.48 words long) than an average document from a Slavic language group (39.18 words long), suggesting that shorter documents are harder to classify by the classifier due to less available information. We can also see that the only group containing documents with similar length as the whole test set are documents containing misleading

[24] Error analysis was conducted on documents misclassified by the system that employed TF-IDF weighting scheme.

named entities (40.96 words long), which suggests that the classifier does somewhat rely on named entities during the prediction process.

5 Cross-lingual news linking

In this section, we compare a number of methods for cross-lingual news linking. We explore some methods not based on topic models and then describe how multilingual topic models can be applied to the task. In Section 5.4, we compare the approaches in the cross-lingual document retrieval (CLDR) task.

5.1 Initial document linking methods

We have explored two cross-lingual document linking methods that are not based on topic models. First is the multilingual embedding-based method where document embeddings are built by taking the sum of the embeddings of the words in the document weighted by frequency. Since the embeddings are multilingual, the resulting document embeddings will also be multilingual. Then to find similar documents across languages, we rank the candidate documents according to their cosine similarity to the query document. This method has been used in (Litschko et al., 2018, 2019; Josifoski et al., 2019).

The next method we explored is the cross-lingual distance metric presented by Balikas et al. (2018). The authors propose the Wasserstein distance to compute distances between documents from different languages. Each document is a set of cross-lingual word embeddings and each word is associated with some weight, such as its term frequency inverse document frequency (TF-IDF). The Wasserstein distance is then the minimum cost of transforming all the words in a query document to the words in a target document. They then demonstrate that using a regularised version of the Wasserstein distance makes the optimisation problem faster to solve and, more importantly, allows multiple associations between words in the query and target documents.

5.2 Topic models

Topic models capture themes inherent in document collections through the co-occurrence patterns of the words in documents. Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is a popular method for inferring these themes or topics. It is generative document model where a document is described by a mixture of different topics and each topic is a probability distribution over the words in the vocabulary. In a document collection we can only observe the words in a document. Therefore, training a model involves inferring these latent variables through approximate inference methods.

A limitation of LDA topic modelling is that it is not applicable to multilingual data. LDA captures co-occurrences of words in documents and words from different languages would rarely, if ever, occur in the same document regardless of their semantics. Multilingual topic models are developed to capture cross-lingual topics from multilingual datasets.

Polylingual Topic Model (PLTM) (Mimno et al., 2009) is a multilingual topic model that extends LDA for an aligned multilingual corpus. Instead of running topic inference on individual documents as in LDA, PLTM infers topics for tuples of documents, where each document in the tuple is in a different language. PLTM assumes that the documents of a tuple discuss the same subject broadly and therefore share the same document-topic distribution.

5.3 Cross-lingual news linking with topic models

In our work on cross-lingual news article linking with topic models, we trained a polylingual topic model (**PLTM**) using a theme-aligned corpora from two languages. After the topic model is trained, we infer document-topic distributions (which we will refer to as the document vector) for unseen articles from both languages. This work has been published in a workshop proceedings (Zosa et al., 2020) and is included in Appendix A.

To find articles in the target language related to a query article in the query language, we take the Jensen-Shannon (JS) divergence between the document-topic distributions of the query article and each of the candidate articles in the target set. The candidate articles are then ranked in ascending order (lower divergence has higher rank) and the top-n ranked articles are returned as the related articles. This approach is similar to what is described in (De Smet & Moens, 2009).

We compared this topic model-based approach with other approaches from literature that use cross-lingual document embeddings (**Cr5**) (Josifoski et al., 2019) and document distance measures (**Wasserstein**) (Balikas et al., 2018).

5.3.1 Dataset

We evaluate using a dataset of Finnish and Swedish news articles published by the Finnish broadcaster YLE and freely available for download from the Finnish Language Bank [25]. The articles are from 2012-18 and are written separately in the two languages (not translations and not parallel). This dataset contains 604,297 articles in Finnish and 228,473 articles in Swedish. Each article is tagged with a set of keywords describing the subject of the article. These keywords were assigned to the articles by a combination of automated methods and manual curation.

To build a topically aligned corpus for training PLTM, we match a Finnish article with a Swedish article if they were published within two days of each other and share three or more keywords. As a result no Finnish article is matched with more than one Swedish article and vice-versa so that we have a set of aligned unique article pairs. We have used this method in the past to train multilingual dynamic topic models (Zosa & Granroth-Wilding, 2019), see Appendix C.

To build a corpus of related news articles for testing, we associate one Finnish article with one or more Swedish articles if they share three or more keywords and if the articles are published in the same month. From this we create three separate test sets: 2013, 2014, and 2015. For each month, we take 100 Finnish articles to use as queries, providing all of the related Swedish articles as a candidate set visible to the models. In this report, we show only the results for the 2013 test set which has 1.3K articles in the candidate set and on average each Finnish article is related to 19.5 Swedish articles (for the complete results and more statistics about the dataset see Appendix A.

5.3.2 Training the PLTM

We use our in-house implementation of PLTM which uses Gibbs sampling for inference. We use 1,000 iterations for burn-in and then infer vectors for unseen documents by sampling every 25th iteration for 200 iterations. To obtain distances between documents, we compute the Jensen-Shannon (JS) divergence between the document-topic distributions of the query document and each of the candidate documents. We trained our model for 100 topics.[26]

[25] `https://www.kielipankki.fi/corpora/`
[26] Source code available on `https://github.com/ezosa/cross-lingual-linking`

Table 8: Precision at k and MRR of cross-lingual linking of related news articles obtained by three stand-alone models and four ensemble models.

Measure:	**P@1**	**P@5**	**P@10**	**MRR**
PLTM	21.8	18.2	16.3	31.6
Wass	21.1	13.7	11.3	30.8
Cr5	32.5	24.5	21.2	41.7
PLTM_Wass	24.6	21.3	19.1	35.2
Cr5_Wass	35.4	27.4	23.2	45.2
PLTM_Cr5	36.4	28.2	24.4	46.6
PLTM_Cr5_Wass	**40.7**	**30.7**	**26.3**	**50.3**

Table 9: Mean Spearman correlation of the ranks of candidate documents for each pair of models.

Model pair	**Correlation**
PLTM, Wass	-0.039
Cr5, Wass	0.128
PLTM, Cr5	0.156

5.4 Results

Table 8 shows the results for each model and ensemble on each of the three test sets, reporting the precision of the top-ranked k results and mean reciprocal rank (MRR). Cr5 is the best-performing stand-alone model by a large margin. Cr5 was originally designed for creating cross-lingual document embeddings by classifying Wikipedia documents according to concepts. We did not retrain it for our particular task. Nevertheless, using these pre-trained word embeddings we were able to retrieve articles that discuss similar subjects in this different domain.

Cr5 outperforms PLTM on its own. One reason may be that 100 topics are too few. We chose this number because it seemed to give topics that are specific enough for short articles but still broad enough that they could reasonably be used to describe similar articles. Another drawback of PLTM is that it does not handle out-of-vocabulary words so there might be significant terms (such as named entities) in the test set that was not part of its training vocabulary and is disregarded during testing.

Wasserstein distance is the worst-performing of the standalone models. A possible reason is that it attempts to transform one document to another and therefore favours documents that share a similar vocabulary to the query document. The technique might be suitable for matching Wikipedia articles (the dataset used in the original paper) because they talk about the same subject at a fine-grained level and use similar words, whilst in our task the goal is to make broader connections between documents.

We created ensemble models by averaging the document distances from the stand-alone models and ranking candidate documents according to this score. We construct four ensemble models by combining each pair of models, as well as all three: **PLTM_Wass**; **Cr5_Wass**; **PLTM_Cr5**; and **PLTM_Cr5_Wass**.

Combining all three models performs best overall. This tells us that each model sometimes finds relevant documents not found by the other models. The correlation of candidate document rankings between the different methods is quite low (see Table 9 and Figure 7). We computed the Spearman correlations between the ranks of the candidate documents produced by each pair of models for each of the queries (query documents) in our test set. As can be seen in the table the correlations are rather low (close to zero for PLTM_Wass and Cr5_Wass), which means that they retrieve documents based on different principles.

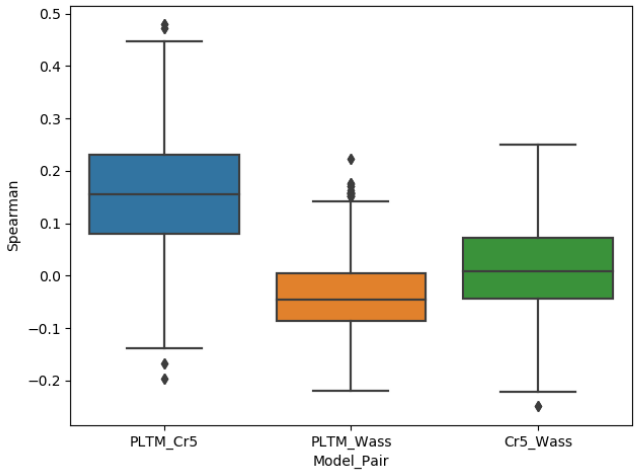


Figure 7: Spearman correlations of the candidate document rankings produced by each pair of models.

5.4.1 Experiments with Embeddia datasets

We experimented with applying PLTM to datasets from some of the Embeddia partners, specifically, the Finnish news articles from STT and Estonian articles from Ekspress Meedia (ExM). Since these articles are not tagged with the same sort of keywords as in the Yle articles that we used to build an aligned corpus, we built an aligned corpus by pairing documents from each language based on the cosine similarity of their cross-lingual document embeddings (see Section 5.1). For this experiment, we included only articles from 2015-2018 and article pairs that have a cosine similarity of more than 0.5. This gave us a corpus size of 31,324 aligned articles. We trained PLTM for 20 topics with this corpus. In Figures 8, 9 and 10, we show some topic word clouds from some of the resulting topics.

6 Associated outputs

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

Description	URL	Availability
Code for ML-DTM	`https://github.com/e/multilingual_dtm`	Public
Code for CLDR evaluation	`https://github.com/ezosa/cross-lingual-linking`	Public
Code for language variety	`http://source.ijs.si/mmartinc/NLE_2017/`	Public
Code for SRNA	`https://gitlab.com/skblaz/srna`	Public
Code for tax2vec	`https://github.com/SkBlaz/tax2vec`	Public

Parts of this work are also described in detail in the following publications.

Figure 8: Topic on the European Union (Left: Estonian, Right: Finnish)

Figure 9: Topic on parliamentary issues (Left: Estonian, Right: Finnish)

Figure 10: Topic on crime and law enforcement (Left: Estonian, Right: Finnish)

Citation	Status	Appendix
Elaine Zosa, Lidia Pivovarova, Mark Granroth-Wilding (2020). A Comparison of Unsupervised Methods for Ad hoc Cross-Lingual Document Retrieval. In proceedings LREC 2020 Workshop on Cross-Language Search and Summarization of Text and Speech. Marseille, France, May 2020.	Published	Appendix A
Matej Martinc and Senja Pollak (2019). Combining n-grams and deep convolutional features for language variety classification. Natural Language Engineering journal.	Published	Appendix B
Elaine Zosa and Mark Granroth-Wilding (2019). Multilingual Dynamic Topic Model. In proceedings Recent Advances in Natural Language Processing (RANLP 2019).	Published	Appendix C
Blaž Škrlj, Jan Kralj, Nada Lavrač, and Senja Pollak (2019). Towards Robust Text Classification with Semantics-Aware Recurrent Neural Architecture. Machine Learning and Knowledge Extraction, 1(2), 575-589.	Published	Appendix D
Blaž Škrlj, Matej Martinc, Jan Kralj, Nada Lavrač, and Senja Pollak (2020). tax2vec: Constructing Interpretable Fea-tures from Taxonomies for Short Text Classification. Computer Speech & Language.	Accepted	Appendix E

7 Conclusion

The focus of this work is on news linking, both on a monolingual and cross-lingual setting, and related techniques to assist with news linking.

In the monolingual setting, we first performed experiments on linking related articles on datasets of 24sata and Vecernji list. We showed that methods such as TF-IDF achieve higher results compared to document embeddings. However, when considering triplets of news manually annotated with semantic similarity, the XLM-RoBERTa model achieved the highest results.

For the related task of *news categorisation*, we presented two methods that allow for incorporating background knowledge, showing that they can improve the classification on short segments of news. In future we will focus on incorporating results of document enrichment methods developed in WP2, while the existing methods can potentially be adapted to a cross-lingual setting. We have also shown that news linking by *language variety* categories achieves very high performance.

In our experiments on cross-lingual document linking, we showed that in retrieving related news articles across languages, a word-embedding based model (Cr5) performed best, followed by the polylingual topic model (PLTM), while the distance-based Wasserstein model has the worst results of the standalone models. We then demonstrated that combining at least two of these methods by averaging their distances yields better results than the models used on their own. Finally, we showed that combining the three models yields the best results. These results indicate that relating documents based on different techniques such as embedding-based or topic-based techniques yields different results and that pooling these results make for a better model.

In future work, in order to integrate the work in this task with other work in the Embeddia project, we will apply the techniques described here to other data received from WPs 1 and 2. This will need to include *events*, *entities* and *keywords*, not just raw text. It may be appropriate when dealing with different types of input data to consider different linking methods, potentially in combination with the text-based linking approaches described here.

Following up on the reported work on monolingual news linking, we will investigate how keyword tagging techniques can help in news topic modelling and categorisation. The methods for keyword extraction,

developed in WP2, have already been applied to the Estonian news dataset. In future, we will see if they can help in improving the results of the methods presented in this deliverable.

With regard to topic model-based linking, we plan to explore embedding-based topic modelling methods such as Gaussian LDA (Das et al., 2015) and Embedded Topic Model (Dieng et al., 2019) and adapt them to cross-lingual settings. Such models could potentially combine the benefits of topic models with word embeddings to retrieve similar documents across languages.

We plan to investigate further ways in which topic modelling can be used for document linking. The approaches described here provide a good starting point. It may be that new ways of using the statistical outputs produced by topic models to draw connections (for example, other vector comparison metrics) can lead to better links, or to find links of a different nature.

Beyond news linking itself, we plan to develop methods for using the results of news linking for practical purposes. We will explore methods for ranking and grouping discovered links, in order to present them to a user in a more useful fashion. We will then explore methods to use the linking techniques to investigate the emergence of *events* over time and to analyse the *spread* of news between places and news sources.

Finally, we will build a news linking system that makes use of all of the above methods that have proved to be of value. This will be applied in real time to linking news data as it appears.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., ... Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems.* (Software available from tensorflow.org)

Ali, A., Dehak, N., Cardinal, P., Khurana, S., Yella, S. H., Glass, J., & Renals, S. (2015). Automatic dialect detection in arabic broadcast speech. In *Proceedings of interspeech.*

Balikas, G., Laclau, C., Redko, I., & Amini, M.-R. (2018). Cross-lingual document retrieval using regularized wasserstein distance. In *European conference on information retrieval* (pp. 398–410).

Basile, P., Caputo, A., & Semeraro, G. (2014). An enhanced lesk word sense disambiguation algorithm through a distributional semantic model. In *Proceedings of coling 2014, the 25th international conference on computational linguistics: Technical papers* (pp. 1591–1600).

Belinkov, Y., & Glass, J. (2016). A character-level convolutional neural network for distinguishing similar languages and dialects. *arXiv preprint arXiv:1609.07568*.

Bestgen, Y. (2017). Improving the character ngram model for the dsl task with bm25 weighting and less frequently used feature sets. In *Proceedings of the fourth workshop on nlp for similar languages, varieties and dialects (vardial)* (pp. 115–123).

Bjerva, J. (2016). Byte-based language identification with deep convolutional networks. *arXiv preprint arXiv:1609.09004*.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Brandes, U. (2001). A faster algorithm for betweenness centrality. *The Journal of Mathematical Sociology*, *25*(2), 163-177.

Chang, C.-C., & Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, *2*(3), 27.

Cho, J., Lee, K., Shin, E., Choy, G., & Do, S. (2015). How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *arXiv preprint arXiv:1511.06348*.

Chollet, F., et al. (2015). Keras: Deep learning library for theano and tensorflow. *URL: https://keras.io/k*, *7*(8), T1.

Clevert, D.-A., Unterthiner, T., & Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of machine learning research*, *12*(Aug), 2493–2537.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., ... Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *ArXiv*, *abs/1911.02116*.

Das, R., Zaheer, M., & Dyer, C. (2015). Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)* (pp. 795–804).

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*(6), 391-407.

De Smet, W., & Moens, M.-F. (2009). Cross-language linking of news stories on the web using interlingual topic modelling. In *Proceedings of the 2nd acm workshop on social web search and mining* (pp. 57–64).

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019, June). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (pp. 4171–4186). Minneapolis, Minnesota: Association for Computational Linguistics. doi: 10.18653/v1/N19-1423

Dieng, A. B., Ruiz, F. J., & Blei, D. M. (2019). Topic modeling in embedding spaces. *arXiv preprint arXiv:1907.04907*.

Fellbaum, C. (Ed.). (1998). *Wordnet: An electronic lexical database*. MIT press.

Gonzalez-Agirre, A., Laparra, E., & Rigau, G. (2012). Multilingual central repository version 3.0: upgrading a very large lexical knowledge base. In *Proceedings of the 6th global wordnet conference (gwc 2012)* (p. online). Matsue.

Goutte, C., Léger, S., & Carpuat, M. (2014). The nrc system for discriminating similar languages. In *Proceedings of the first workshop on applying nlp tools to similar languages, varieties and dialects* (pp. 139–145).

Grässer, F., Kallumadi, S., Malberg, H., & Zaunseder, S. (2018). Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. In *Proceedings of the 2018 international conference on digital health* (pp. 121–125). New York, NY, USA: ACM.

Greene, D., & Cunningham, P. (2006). Practical solutions to the problem of diagonal dominance in kernel document clustering. In *Proceedings of the 23rd international conference on machine learning (icml'06)* (pp. 377–384). ACM Press.

Josifoski, M., Paskov, I. S., Paskov, H. S., Jaggi, M., & West, R. (2019). Crosslingual document embedding as reduced-rank ridge regression. In *Proceedings of the twelfth acm international conference on web search and data mining* (pp. 744–752).

Jursic, M., Mozetic, I., Erjavec, T., & Lavrac, N. (2010). Lemmagen: Multilingual lemmatisation with induced ripple-down rules. *J. UCS*, *16*(9), 1190–1214. doi: 10.3217/jucs-016-09-1190

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kralj, J. (2017). *Heterogeneous information network analysis for semantic data mining: Doctoral dissertation* (Unpublished doctoral dissertation).

Kralj, J., Robnik-Šikonja, M., & Lavrač, N. (2019). NetSDM: Semantic data mining with network analysis. *Journal of Machine Learning Research*, *20*(32), 1-50.

Lazer, D. M., Baum, M. A., Benkler, Y., Berinsky, A. J., Greenhill, K. M., Menczer, F., ... others (2018). The science of fake news. *Science*, *359*(6380), 1094–1096.

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *International conference on machine learning* (pp. 1188–1196).

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436.

Litschko, R., Glavaš, G., Ponzetto, S. P., & Vulić, I. (2018). Unsupervised cross-lingual information retrieval using monolingual data only. In *The 41st international acm sigir conference on research & development in information retrieval* (pp. 1253–1256).

Litschko, R., Glavaš, G., Vulic, I., & Dietz, L. (2019). Evaluating resource-lean cross-lingual embedding models in unsupervised retrieval. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval* (pp. 1109–1112).

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*.

Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A., & Tiedemann, J. (2016). Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. In *Proceedings of the third workshop on nlp for similar languages, varieties and dialects (vardial3)* (pp. 1–14).

Martinc, M., & Pollak, S. (2019). Combining n-grams and deep convolutional features for language variety classification. *Natural Language Engineering*, *25*(5), 607–632.

Martinc, M., Škrjanec, I., Zupan, K., & Pollak, S. (2017). Pan 2017: Author profiling - gender and language variety prediction. In *Clef* (p. online).

Meyer, D., Leisch, F., & Hornik, K. (2003). The support vector machine under test. *Neurocomputing*, *55*(1), 169 - 186. (Support Vector Machines) doi: https://doi.org/10.1016/S0925-2312(03)00431-4

Miller, G. A. (1995, November). Wordnet: A lexical database for english. *Commun. ACM*, *38*(11), 39–41.

Mimno, D., Wallach, H. M., Naradowsky, J., Smith, D. A., & McCallum, A. (2009). Polylingual topic models. In *Proceedings of the 2009 conference on empirical methods in natural language processing: Volume 2-volume 2* (pp. 880–889).

Myers, I. B. (1962). The myers-briggs type indicator: Manual (1962).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... others (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, *12*, 2825–2830.

Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on pattern analysis and machine intelligence*, *27*(8), 1226–1238.

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)* (pp. 1532–1543).

Rangel, F., Rosso, P., Chugur, I., Potthast, M., Trenkmann, M., Stein, B., ... Daelemans, W. (2014). Overview of the 2nd author profiling task at PAN 2014. In *Working notes papers of the clef conference* (pp. 1–30).

Rangel, F., Rosso, P., Potthast, M., & Stein, B. (2017). Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. *Working Notes Papers of the CLEF*, 1613–0073.

Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M., & Stein, B. (2016). Overview of the 4th author profiling task at pan 2016: cross-genre evaluations. In *Working notes papers of the clef 2016 evaluation labs. ceur workshop proceedings/balog, krisztian [edit.]; et al.* (pp. 750–784).

Řehůřek, R., & Sojka, P. (2010, May 22). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks* (pp. 45–50). Valletta, Malta: ELRA. (`http://is.muni.cz/publication/884893/en`)

Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *Emnlp/ijcnlp.*

Robertson, S., & Zaragoza, H. (2009). *The probabilistic relevance framework: Bm25 and beyond*. Now Publishers Inc.

Samardzic, T., Scherrer, Y., & Glaser, E. (2016). Archimob-a corpus of spoken swiss german. In *Proceedings of lrec 2016* (p. 4061–4066).

Škrlj, B., Martinc, M., Kralj, J., Lavrač, N., & Pollak, S. (2020). tax2vec: Constructing interpretable features from taxonomies for short text classification. *Computer Speech & Language*, 101104.

Tan, L., Zampieri, M., Ljubešic, N., & Tiedemann, J. (2014). Merging comparable data sources for the discrimination of similar languages: The dsl corpus collection. In *Proceedings of the 7th workshop on building and using comparable corpora (bucc)* (pp. 11–15).

Tan, L., Zampieri, M., Ljubešic, N., & Tiedemann, J. (2014). Merging comparable data sources for the discrimination of similar languages: The dsl corpus collection. In *Proceedings of the 7th workshop on building and using comparable corpora (bucc)* (p. 11-15).

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 30* (pp. 5998–6008).

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, *abs/1910.03771*.

Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 1480–1489).

Zampieri, M., Malmasi, S., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J., ... Aepli, N. (2017). Findings of the vardial evaluation campaign 2017.

Zampieri, M., Tan, L., Ljubešić, N., Tiedemann, J., & Nakov, P. (2015). Overview of the dsl shared task 2015. In *Proceedings of the joint workshop on language technology for closely related languages, varieties and dialects* (pp. 1–9).

Zosa, E., & Granroth-Wilding, M. (2019). Multilingual dynamic topic model. In *Proceedings recent advances in natural language processing 2019.*

Zosa, E., Granroth-Wilding, M., & Pivovarova, L. (2020). A comparison of unsupervised methods for ad hoc cross-lingual document retrieval. In *Proceedings of the lrec 2020 workshop on cross-language search and summarization of text and speech.*

Škrlj, B., Kralj, J., Lavrač, N., & Pollak, S. (2019, Apr). Towards robust text classification with semantics-aware recurrent neural architecture. *Machine Learning and Knowledge Extraction*, *1*(2), 575–589. Retrieved from `http://dx.doi.org/10.3390/make1020034` doi: 10.3390/make1020034

Appendix A: A Comparison of Unsupervised Methods for Ad hoc Cross-Lingual Document Retrieval

A Comparison of Unsupervised Methods for Ad hoc Cross-Lingual Document Retrieval

Elaine Zosa, Mark Granroth-Wilding, Lidia Pivovarova
University of Helsinki
Helsinki, Finland
firstname.lastname@helsinki.fi

Abstract

We address the problem of linking related documents across languages in a multilingual collection. We evaluate three diverse unsupervised methods to represent and compare documents: (1) multilingual topic model; (2) cross-lingual document embeddings; and (3) Wasserstein distance. We test the performance of these methods in retrieving news articles in Swedish that are known to be related to a given Finnish article. The results show that ensembles of the methods outperform the stand-alone methods, suggesting that they capture complementary characteristics of the documents.

1. Introduction

We address the problem of retrieving related documents across languages through unsupervised cross-lingual methods that do not use translations or other lexical resources, such as dictionaries. There is a multitude of multilingual resources on the Internet such as Wikipedia, multilingual news sites, and historical archives. Many users may speak multiple languages or work in a context where discovering related documents in different languages is valuable, such as historical enquiry. This calls for tools that relate resources across language boundaries.

We choose to focus on methods that do not use translations because lexical resources and translation models vary across languages and time periods. Our goal is to find methods that are applicable across these contexts without extensive fine-tuning or manual annotation. Much work on cross-lingual document retrieval (CLDR) has focused on *cross-lingual word embeddings* but topic-based methods have also been used (Wang et al., 2016). Previous work has applied such cross-lingual learning methods to *known item search* where the task is to retrieve one relevant document given a query document (Balikas et al., 2018; Josifoski et al., 2019; Litschko et al., 2019). We are interested in *ad hoc retrieval* where there could be any number of relevant documents and the task is to rank the documents in the target collection according to their relevance to the query document (Voorhees, 2003).

Here we evaluate three existing unsupervised or weakly supervised methods previously used in CLDR for slightly different tasks: (1) multilingual topic model (MLTM); (2) document embeddings derived from cross-lingual reduced rank ridge regression or Cr5 (Josifoski et al., 2019) and; (3) Wasserstein distance for CLDR (Balikas et al., 2018). These methods link documents across languages in fundamentally different ways. MLTM induces a shared cross-lingual topic space and represents documents as a language-independent distribution over these topics; Cr5 obtains cross-lingual document embeddings; and the Wasserstein distance as used by (Balikas et al., 2018) computes distances between documents as sets of cross-lingual word embeddings (Speer et al., 2016). The methods broadly cover the landscape of recent CLDR methods. To our knowledge, this is the first comparison of Cr5 and Wasserstein for ad hoc retrieval.

This paper adds to the literature on CLDR in three ways: (1) evaluating unsupervised methods for retrieving related documents across languages (ad hoc retrieval), in contrast to retrieval of a single corresponding document; (2) evaluating different ensembling methods; and (3) demonstrating the effectiveness of relating documents across languages through complementary methods.

2. Related Work

Previous work on linking documents across languages has used translation-based features, where the query is translated into the target language and the retrieval task proceeds in the target language (Hull and Grefenstette, 1996; Litschko et al., 2018; Utiyama and Isahara, 2003). Other methods used term-frequency correlation (Tao and Zhai, 2005; Vu et al., 2009), sentence alignment (Utiyama and Isahara, 2003), and named entities (Montalvo et al., 2006). In this paper, we are interested in language-independent models with minimal reliance on lexical resources and other metadata or annotations.

2.1. Multilingual topic model

The multilingual topic model (MLTM) is an extension of LDA topic modelling (Blei et al., 2003) for comparable multilingual corpora (De Smet and Moens, 2009; Mimno et al., 2009). In contrast to LDA, which learns topics by treating each document as independent, MLTM relies on a topically aligned corpus, which consists of tuples of documents in different languages discussing the same themes. MLTM learns separate but aligned topic distributions over the vocabularies of the languages represented in the corpus. One of the main advantages of MLTM is that it can extend across any number of languages, not just two, as long as there is a topically aligned corpus covering these languages. This can be difficult because aligning corpora is not a trivial task, especially as the number of languages gets larger. For this reason, Wikipedia, currently in more than 200 languages, is a popular source of training data for MLTM.

Another issue facing topic models is that the choice of hyperparameters can significantly affect the quality and nature of topics extracted from the corpus and, consequently,

its performance in the downstream task we want use it for. There are three main hyperparameters in LDA-based models: the number of topics to extract, K; the document concentration parameter, α, that controls the sparsity of the topics associated with each document; and the topic concentration parameter, β, which controls the sparsity of the topic-specific distribution over the vocabulary.

2.2. Cross-lingual document embeddings

Cross-lingual reduced-rank ridge regression (Cr5) was recently introduced as a novel method of obtaining cross-lingual document embeddings (Josifoski et al., 2019). The authors formulate the problem of inducing a shared document embedding space as a linear classification problem. Documents in a multilingual corpus are assigned language-independent concepts. The linear classifier is trained to assign the concepts to documents, learning a matrix of weights W that embeds documents in a concept space close to other documents labelled with the same concept and far from documents expressing different concepts.
They train on a multilingual Wikipedia corpus, where articles are assigned labels based on language-independent Wikipedia concepts. They show that the method outperforms the state-of-the-art cross-lingual document embedding method from previous literature (Litschko et al., 2018). Cr5 is trained to produce document embeddings, but can also be used to obtain embeddings for smaller units, such as sentences and words. One disadvantage is that it requires labelled documents for training. However, the induced cross-lingual vectors can then be used for any tasks in which the input document is made up of words in the vocabulary of the corresponding language in the training set.

2.3. Wasserstein distances for documents

Wasserstein distance is a distance metric between probability distributions and has been previously used to compute distances between text documents in the same language (*Word Mover's Distance* (Kusner et al., 2015)). In (Balikas et al., 2018) the authors propose the Wasserstein distance to compute distances between documents from different languages. Each document is a set of cross-lingual word embeddings (Speer et al., 2016) and each word is associated with some weight, such as its term frequency inverse document frequency (tf.idf). The Wasserstein distance is then the minimum cost of transforming all the words in a query document to the words in a target document. They then demonstrate that using a regularized version of the Wasserstein distance makes the optimization problem faster to solve and, more importantly, allows multiple associations between words in the query and target documents.

3. Experimental setup

3.1. Task and dataset

We evaluate using a dataset of Finnish and Swedish news articles published by the Finnish broadcaster YLE and freely available for download from the Finnish Language Bank[1]. The articles are from 2012-18 and are written separately in the two languages (not translations and not parallel). This dataset contains 604,297 articles in Finnish and 228,473 articles in Swedish. Each article is tagged with a set of keywords describing the subject of the article. These keywords were assigned to the articles by a combination of automated methods and manual curation. The keywords vary in specificity, from named entities, such as *Sauli Niinisto* (the Finnish president), to general subjects, such as *talous* (sv: *ekonomi*, en: economy). On average, Swedish articles are tagged with five keywords and 15 keywords for Finnish articles. Keywords are provided in Finnish and Swedish regardless of the article language so no additional mapping is required.

	MLTM Train set	Test set	
	articles per lang	**#candidates**	**#related**
2012	7.2K	-	-
2013	7.2K	1.3K	19.5
2014	7.2K	1.4K	31.8
2015	-	1.5K	35.9

Table 1: Statistics of the training set for training MLTMs and test sets for each year. #candidates is the average size of the candidate articles set and #related is the average number of Swedish articles related to each Finnish article.

To build a corpus of related news articles for testing, we associate one Finnish article with one or more Swedish articles if they share three or more keywords and if the articles are published in the same month. From this we create three separate test sets: 2013, 2014, and 2015. For each month, we take 100 Finnish articles to use as queries, providing all of the related Swedish articles as a candidate set visible to the models.
To build a topically aligned corpus for training MLTM, we match a Finnish article with a Swedish article if they were published within two days of each other and share three or more keywords. As a result no Finnish article is matched with more than one Swedish article and vice-versa so that we have a set of aligned unique article pairs. To train MLTM we use a year which is preceding the testing year: e.g., we train a model using articles from 2012 and test it on articles from 2013. Unaligned articles are not used for either training or testing. The script for article alignment will be provided in the Github repository for this work.
Table 1 shows the statistics of the training and test sets. As can be seen in the last column of the table, one Finnish article corresonds to almost twenty Swedish articles for the 2013 dataset and more than thirty for the other two datasets. This is typical for large news collections, since one article may have an arbitrary number of related articles. Thus, our corpus is more suitable for ad-hoc search evaluation than Wikipedia or Europarl corpus, since they contain only one-to-one relation[2].

3.2. Models

We use our in-house implementation of MLTM training using Gibbs sampling[3]. The training corpus was tokenized, lemmatized and stopwords were removed. We limited the

[1] https://www.kielipankki.fi/corpora/

[2] CLEF 2000-2003 ad-hoc retrieval Test Suite, which also contains many-to-many relations, is not freely available

[3] https://github.com/ezosa/cross-lingual-linking.git

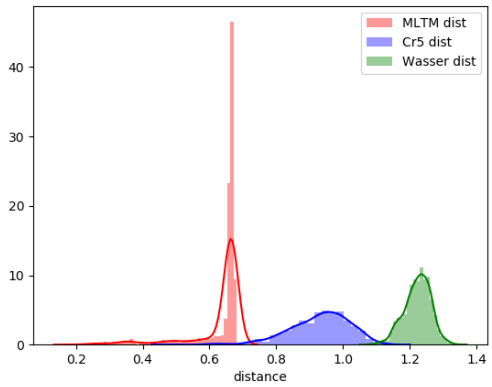


Figure 1: Density plots of the distances between one query document and the candidate documents.

vocabulary to the 9,000 most frequent terms for each language. We train three separate models for 2012, 2013, and 2014 (for the 2013, 2014, and 2015 test sets, respectively). We train all three models with $K = 100$ topics, $\alpha = 1/K$ and $\beta = 0.08$. We use 1,000 iterations for burn-in and then infer vectors for unseen documents by sampling every 25th iteration for 200 iterations. To obtain distances between documents, we compute the Jensen-Shannon (JS) divergence between the document-topic distributions of the query document and each of the candidate documents.

For Cr5, we use pretrained word embeddings for Finnish and Swedish provided by the authors[4]. We construct document embeddings according to the original method – by summing up the embeddings of the words in the document weighted by their frequency. We compute the distance between documents as the cosine distance of the document embeddings.

For Wasserstein distance, we use code provided by the authors for computing distances between documents and use the same cross-lingual embeddings they did in their experiments[5] (Speer et al., 2016). Wasserstein distance has a regularization parameter λ that controls how the model matches words in the query and candidate documents. The authors suggested using $\lambda = 0.1$ because it encourages more relaxed associations between words. Higher values of λ create stronger associations while too low values fail to associate words that are direct translations of each other. In this task, it might make more sense to use lower λ values, though an experiment with $\lambda = 0.01$ brought no noticeable improvement in performance (see Section 3.3.).

We created ensemble models by averaging the document distances from the stand-alone models and ranking candidate documents according to this score. We construct four ensemble models by combining each pair of models, as well as all three: **MLTM_Wass**; **Cr5_Wass**; **MLTM_Cr5**; and **MLTM_Cr5_Wass**.

3.3. Results and Discussion

Table 2 shows the results for each model and ensemble on each of the three test sets, reporting the precision of the top-ranked k results and mean reciprocal rank (MRR). Cr5 is the best-performing stand-alone model by a large margin. Cr5 was originally designed for creating cross-lingual document embeddings by classifying Wikipedia documents according to concepts. We did not retrain it for our particular task. Nevertheless, using these pre-trained word embeddings we were able to retrieve articles that discuss similar subjects in this different domain. However, it is worth noting that Cr5 can only be trained on languages for which labels are available for *some* similarly transferable training domain.

MLTM, being a topic-based model, would seem like the obvious choice for a task like this because we want to find articles that share some broad characteristics with the query document, even if they do not discuss the same named entities or use similar words. However, Cr5 outperforms MLTM on its own. One reason may be that 100 topics are too few. We chose this number because it seemed to give topics that are specific enough for short articles but still broad enough that they could reasonably be used to describe similar articles. Another drawback of this model is that it does not handle out-of-vocabulary words and the choice of using a vocabulary of 9,000 terms might be too low.

Wasserstein distance is the worst-performing of the stand-alone models especially for the 2014 and 2015 test sets where it offers little improvement when ensembled with Cr5 (Cr5_Wass). A possible reason is that it attempts to transform one document to another and therefore favors documents that share a similar vocabulary to the query document. The technique might be suitable for matching Wikipedia articles, as shown in (Balikas et al., 2018) because they talk about the same subject at a fine-grained level and use similar words, whilst in our task the goal is to make broader connections between documents.

In Figure 1, the density plots of the distances of one query document and the candidate documents. We see that MLTM and Wasserstein tend to have sharper peaks while Cr5 distances are flatter. MLTM has minimum and maximum distances of 0.2 and 0.68, respectively, while Cr5 has 0.49 and 1.14, and Wasserstein has 1.08 and 1.34. Topic modelling tends to predict that most of the target documents are far from the query document (peaks at the right side). This is not only true for this particular query document but for other query documents in our test set as well. We also see that Wasserstein has larger distances which is potentially problematic. We tried normalizing the distances produced by the models such that they are centered at zero and using these distances for the ensembled model however it produces the same document rankings as the unnormalized distances. This might be because we are only concerned with the documents with the smallest distances where Wasserstein does not contribute much.

For the ensemble models, combining all three models per-

[4]https://github.com/epfl-dlab/Cr5

[5]https://github.com/balikasg/WassersteinRetrieval

Test set:	2013				2014				2015			
Measure:	**P@1**	**P@5**	**P@10**	**MRR**	**P@1**	**P@5**	**P@10**	**MRR**	**P@1**	**P@5**	**P@10**	**MRR**
MLTM	21.8	18.2	16.3	31.6	24.1	22.4	20.6	34.8	30.8	29.0	27.1	41.6
Wass	21.1	13.7	11.3	30.8	21.0	16.9	14.7	31.9	25.1	20.6	17.9	37.2
Wass $\lambda = 0.01$	20.3	13.5	11.1	30.0	21.3	16.8	14.6	32.0	25.1	20.1	17.3	36.6
Cr5	32.5	24.5	21.2	41.7	38.3	30.2	26.0	48.0	43.1	37.1	33.5	53.8
MLTM_Wass	24.6	21.3	19.1	35.2	27.3	25.5	23.4	38.2	30.4	31.4	30.1	42.9
Cr5_Wass	35.4	27.4	23.2	45.2	38.1	32.2	28.2	49.2	41.2	37.7	34.9	52.9
MLTM_Cr5	36.4	28.2	24.4	46.6	**44.8**	34.3	30.1	53.6	42.7	40.1	36.9	54.5
MLTM_Cr5_Wass	**40.7**	**30.7**	**26.3**	**50.3**	43.0	**36.1**	**31.9**	**53.8**	**44.5**	**41.3**	**38.5**	**55.9**

Table 2: Precision at k and MRR of cross-lingual linking of related news articles obtained by three stand-alone models and four ensemble models.

Test set:	2013	2014	2015	AVG
MLTM, Wass	-0.039	-0.016	-0.022	-0.026
Cr5, Wass	0.128	0.027	0.026	0.060
MLTM, Cr5	0.156	0.164	0.178	0.166

Table 3: Mean Spearman correlation of the ranks of candidate documents for each pair of models.

forms best overall for all three test sets and all but one precision level—the only exception is P1 for 2014 where MLTM_Cr5 achieves roughly the same performance. This tells us that each model sometimes finds relevant documents not found by the other models. The correlation of candidate document rankings between the different methods is quite low (Table 3). We compute the correlation between the ranks for each of the 1200 query documents (100 queries for each month) for each year of our test set and average them. As can be seen in the table the correlations are rather low, which means that they retrieve documents based on different principles. The highest correlation is between **MLTM** has the **Cr5** while correlation between **MLTM** and **Wass** is the lowest.

This suggests that there are different ways of retrieving related documents across languages and that the three methods of cross-lingual embeddings, cross-lingual topic spaces and cross-lingual distance measures capture complementary notions of similarity. A simple combination of their decisions is thus able to make better judgements than any can make on its own.

As an example, in Table 4 we show excerpts from a query article in Finnish and some of the related Swedish articles correctly predicted by the different models. For this article, Cr5 gave 10 correct predictions in its top 10 (perfect precision), MLTM gave 8 correct predictions and Wasserstein only 4. Like Cr5, the ensemble model MLTM_Cr5_Wass also achieved perfect precision. MLTM and MLTM_Cr5_Wass shared 4 correct predictions while Cr5 and MLTM_Cr5_Wass shared 7. All the articles correctly predicted by Wasserstein were also predicted by the other models. We show articles from Cr5, MLTM and MLTM_Cr5_Wass that was correctly predicted by that model only and for Wasserstein, we show the top correct article that it predicted.

4. Conclusions and Future work

In this paper we compare three different methods for cross-lingual ad hoc document retrieval by applying them to the task of retrieving Swedish news articles that are related to a given Finnish article. We show that a word-embedding based model, Cr5, performs best followed by the multilingual topic model and the distance-based Wasserstein model has the worst results of the stand-alone models. We then demonstrate that combining at least two of these methods by averaging their distances yields better results than the models used on their own. Finally we show that combining the three models yields the best results. These results tell us that relating documents based on different techniques such as embedding-based or topic-based techniques yields different results and that pooling these results make for a better model.

In the future we plan to investigate the performance of word embedding-based multilingual topic models in this task. There is already some work done on developing topic models that use word embeddings (Batmanghelich et al., 2016; Das et al., 2015). To our knowledge, they have not yet been applied to cross-lingual embeddings. Such a model could potentially combine the benefits of the multilingual topic model with word embeddings for retrieving similar documents across languages.

We also plan to further experiments with multilingual topic models for languages where the amount of linked documents is scarce. In this work, we trained the topic model with thousands of linked articles because the articles were annotated with tags however this might not always be the case, for instance with historical data sets or under-resourced languages where there are not readily available annotated data and manual annotation is time-consuming or requires expert knowledge. In such cases, we could still train a multilingual topic model with smaller amounts of aligned training data or perhaps a training set where some articles do not have a counterpart article in the other language.

There is also scope for further exploration of ensemble methods, going beyond the simple combination of distance metrics we have applied here. As well as combining models in different ways, further, potentially complementary,

Query article	Yleisradion YleX-kanavan kymmenen suosituimman kappaleen listalla,valtaosa on suomalaisartisteja tai -yhtyeitä. Radio Suomen kaikki,kymmenen eniten kuultua kappaletta ovat odotetusti kotimaisia. YleX ja Radio Suomi ovat koonneet listan eniten soittamastaan musiikista vuonna 2012.
MLTM	På min låtlista finns låtar som på olika sätt och från olika perspektiv beskriver livets grundläggande vemod eller ”life bitter-sweet”, som man brukar säga på Irland. Det säger Tom Sjöblom, som har valt musiken denna vecka i [Min musik.]
Cr5	De isländska banden tar över världen, vi träffade Sóley som nyligen varit på USA-turné med sina isländska kollegor Of Monsters And Men. **Sóley** är isländska och betyder solros. Sóley är också namnet på sångerskan som är en av de mest intressanta nya musikexporterna som kommit från Island.
Wasserstein	Både Radio Vega och Radio Extrem har börjat spela låtar som tävlar i Tävlingen för ny musik UMK. Radio Extrem har tagit in både Krista Siegfrids __Marry me__ och Diandras __Colliding into you__ på spellistan, och låtarna kommer att spelas två gånger om dagen åtminstone nu i början.
MLTM_Cr5_Wass	Smakproven på 30 sekunder av de tolv UMK låtarna kittlade fantasin så,där passligt, men nu behöver vi inte längre gissa oss till hur sångerna,låter i sin helhet. De färdigt producerade bidragen kan nu höras på,Arenan.

Table 4: Excerpt from a query Finnish article and some related Swedish articles correctly predicted by the models. The query article is about popular songs on Finnish radio.

measures of document similarity could be included: for example, explicitly taking into account overlap of named entities, or document publishing metadata if such information is available.

Acknowledgements

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant 770299 (NewsEye) and 825153 (EMBEDDIA).

References

Balikas, G., Laclau, C., Redko, I., and Amini, M.-R. (2018). Cross-lingual document retrieval using regularized Wasserstein distance. In *European Conference on Information Retrieval*, pages 398–410. Springer.

Batmanghelich, K., Saeedi, A., Narasimhan, K., and Gershman, S. (2016). Nonparametric spherical topic modeling with word embeddings. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 2016, page 537. NIH Public Access.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Das, R., Zaheer, M., and Dyer, C. (2015). Gaussian LDA for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795–804.

De Smet, W. and Moens, M.-F. (2009). Cross-language linking of news stories on the web using interlingual topic modelling. In *Proceedings of the 2nd ACM workshop on Social web search and mining*, pages 57–64. ACM.

Hull, D. A. and Grefenstette, G. (1996). Querying across languages: a dictionary-based approach to multilingual information retrieval. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–57. Citeseer.

Josifoski, M., Paskov, I. S., Paskov, H. S., Jaggi, M., and West, R. (2019). Crosslingual document embedding as reduced-rank ridge regression. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 744–752. ACM.

Kusner, M., Sun, Y., Kolkin, N., and Weinberger, K. (2015). From word embeddings to document distances. In *International conference on machine learning*, pages 957–966.

Litschko, R., Glavaš, G., Ponzetto, S. P., and Vulić, I. (2018). Unsupervised cross-lingual information retrieval using monolingual data only. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1253–1256. ACM.

Litschko, R., Glavaš, G., Vulic, I., and Dietz, L. (2019). Evaluating resource-lean cross-lingual embedding models in unsupervised retrieval. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1109–1112. ACM.

Mimno, D., Wallach, H. M., Naradowsky, J., Smith, D. A., and McCallum, A. (2009). Polylingual topic models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 880–889. Association for Computational Linguistics.

Montalvo, S., Martinez, R., Casillas, A., and Fresno, V. (2006). Multilingual document clustering: an heuristic approach based on cognate named entities. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1145–1152. Association for Computational Linguistics.

Speer, R., Chin, J., and Havasi, C. (2016). ConceptNet 5.5: An open multilingual graph of general knowledge. *CoRR*, abs/1612.03975.

Tao, T. and Zhai, C. (2005). Mining comparable bilingual text corpora for cross-language information integration. In *Proceedings of the eleventh ACM SIGKDD interna-*

tional conference on Knowledge discovery in data mining, pages 691–696. ACM.

Utiyama, M. and Isahara, H. (2003). Reliable measures for aligning Japanese-English news articles and sentences. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 72–79. Association for Computational Linguistics.

Voorhees, E. (2003). Overview of TREC 2003. pages 1–13, 01.

Vu, T., Aw, A. T., and Zhang, M. (2009). Feature-based method for document alignment in comparable news corpora. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 843–851. Association for Computational Linguistics.

Wang, Y.-C., Wu, C.-K., and Tsai, R. T.-H. (2016). Cross-language article linking with different knowledge bases using bilingual topic model and translation features. *Knowledge-Based Systems*, 111:228–236.

Appendix B: Combining n-grams and deep convolutional features for language variety classification

Natural Language Engineering (2019), **25**, pp. 607–632
doi:10.1017/S1351324919000299

CAMBRIDGE UNIVERSITY PRESS

ARTICLE

Combining *n*-grams and deep convolutional features for language variety classification

Matej Martinc[1]* and Senja Pollak[1,2]

[1]Department of Knowledge Technologies, Jožef Stefan Institute, Ljubljana, Slovenia and
[2]Usher Institute of Population Health Sciences and Informatics, Edinburgh Medical School, Usher Institute, University of Edinburgh, Edinburgh, UK
*Corresponding author. Email: matej.martinc@ijs.si

(Received 05 November 2018; revised 12 April 2019; accepted 12 May 2019; first published online 18 July 2019)

Abstract
This paper presents a novel neural architecture capable of outperforming state-of-the-art systems on the task of language variety classification. The architecture is a hybrid that combines character-based convolutional neural network (CNN) features with weighted bag-of-*n*-grams (BON) features and is therefore capable of leveraging both character-level and document/corpus-level information. We tested the system on the Discriminating between Similar Languages (DSL) language variety benchmark data set from the VarDial 2017 DSL shared task, which contains data from six different language groups, as well as on two smaller data sets (the Arabic Dialect Identification (ADI) Corpus and the German Dialect Identification (GDI) Corpus, from the VarDial 2016 ADI and VarDial 2018 GDI shared tasks, respectively). We managed to outperform the winning system in the DSL shared task by a margin of about 0.4 percentage points and the winning system in the ADI shared task by a margin of about 0.2 percentage points in terms of weighted F1 score without conducting any language group-specific parameter tweaking. An ablation study suggests that weighted BON features contribute more to the overall performance of the system than the CNN-based features, which partially explains the uncompetitiveness of deep learning approaches in the past VarDial DSL shared tasks. Finally, we have implemented our system in a workflow, available in the ClowdFlows platform, in order to make it easily available also to the non-programming members of the research community.

Keywords: language variety; author profiling; text classification; convolutional neural network; bag-of-n-grams

1. Introduction

Author profiling (AP), which deals with learning about the demographics of a person based on the text she or he produced, is becoming a strong trend in the field of natural language processing (NLP). Tasks such as age, gender, and language variety prediction (automatic distinction between similar dialects or languages) are becoming increasingly popular, in part also because of the marketing potential of this research. Most AP research communities are centered around a series of scientific events and shared tasks on digital text forensics, the two most popular being the evaluation campaign VarDial (Varieties and Dialects)[a] (Zampieri *et al.* 2014), focused on tasks related to the study of linguistic variation, and an event called PAN (Uncovering Plagiarism, Authorship, and Social Software Misuse)[b], which first took place in 2011 and was followed by a series of shared tasks organized since 2013 (Rangel *et al.* 2013).

[a]http://corporavm.uni-koeln.de/vardial/sharedtask.html
[b]http://pan.webis.de/

Table 1. Winning systems for AP classification tasks in PAN AP and VarDial DSL shared tasks (language variety tasks in bold)

Year	VarDial (DSL – closed track)	PAN (AP)
2014	**SVM + BON** (Goutte, Léger, and Carpuat (2014))	LIBLINEAR[5] + BON (López-Monroy *et al.* 2014)
2015	**SVM + BON** (Malmasi and Dras 2015)	LIBLINEAR[5] + BON (Alvarez-Carmona *et al.* 2015)
2016	**SVM + BON** (Çöltekin and Rama 2016)	SVM + BON (Vollenbroek *et al.* 2016)
2017	**SVM + BON** (Bestgen 2017)	**SVM + BON** (Basile *et al.* 2017)

While deep learning approaches are gradually taking over different areas of NLP, the best approaches to AP still use more traditional classifiers and require extensive feature engineering (Rangel, Rosso, Potthast *et al.* 2017). This fact can be clearly seen if we look at the architectures used by the teams winning the AP shared tasks in recent years. Table 1 presents the winning approaches to the VarDial Discriminating between Similar Languages(DSL) shared tasks and PAN AP (gender, age, personality, and language variety prediction) tasks between 2014 and 2017[c]. In fact, six out of eight winning teams used one or an ensemble of Support Vector machine (SVM) classifiers and bag-of-*n*-grams (BON) features[d] for classification (two other winning teams used a LIBLINEAR classifier[e] and BON features), and when it comes to the task of DSL (all VarDial DSL tasks and PAN 2017 AP task), SVM classifiers with BON features have been used by all of the winning teams. The best ranking system that employed a deep learning architecture was developed by Miura *et al.* (2017) and ranked fourth in the PAN 2017 AP shared task.

The main contribution of this paper is to demonstrate that it is possible to build a neural architecture capable of achieving state-of-the-art results in the field of AP, and more specifically on the task of DSL. The proposed neural system is unique in a sense that it combines sophisticated feature engineering techniques used in traditional approaches to text classification with the newer neural automatic feature construction in order to achieve synergy between these two feature types. Experiments were conducted on eight distinct language varieties. First, we report results on the DSL Corpus Collection (DSLCC) v4.0 (Tan *et al.* 2014) used in VarDial 2017 (Zampieri *et al.* 2017), which was chosen because of its size (with 294,000 documents it is by far the largest corpus used in the presented shared tasks) and because it contains six different language groups, which also allows to explore the possibility of building a generic architecture that would discriminate well between languages in many different language groups without any language-group-specific parameter tweaking. Second, we report results on two much smaller corpora, the Arabic Dialect Identification Corpus (ADIC) used in a VarDial 2016 ADI shared task (Malmasi *et al.* 2016*b*) and the German Dialect Identification Corpus (GDIC) used in a VarDial 2018 GDI shared task (Zampieri *et al.* 2018) in order to determine how data set size and characteristics affect the competitiveness of the proposed system. Finally, we want to encourage the reproducibility of results and offer a larger research community (including linguists and social scientists) an easy out-of-the-box way of using our system. Therefore, we have not only published our code online (`http://source.ijs.si/mmartinc/NLE_2017`) but also implemented the architecture in the clowd-based visual programming system ClowdFlows (Kranjc, Podpečan, and Lavrač (2012)).

[c]VarDial evaluation campaign 2018 was not included because there was no DSL shared task. PAN 2018 gender classification task is not included because the gender classification task dealt with determining the gender of the author from both text and image data.

[d]The term BON features is used in a broader sense here, covering features such as bag-of-words, character, and word BON and bag-of-part-of-speech tags.

[e]It is unclear from the system description papers by López-Monroy *et al.* (2014) and Alvarez-Carmona *et al.* (2015) whether linear SVM or logistic regression classifier was used.

The paper is structured as follows. Section 2 addresses the related work on text classification in the field of AP. Section 3 describes the architecture of the proposed neural classification system in detail, while in Section 4 we report on our experimental setup. Results of the experiments and an error analysis are presented in Section 5, followed by an ablation study in Section 6. Section 7 presents the implementation of our approach in the ClowdFlows platform and finally, the conclusions and directions for further work are presented in Section 8.

2. Related work

The most popular approach to language variety classification usually relies on BON features and SVM classifiers (see Table 1). Bestgen (2017), the winner of the VarDial 2017 DSL task, used an SVM classifier with character *n*-grams, capitalized word character *n*-grams, *n*-grams of part-of-speech (POS) tags, and global statistics (proportions of capitalized letters, punctuation marks, spaces, etc.) features. *N*-grams had sizes from one to seven and different feature configurations were used for different language groups. The novelty of this approach was the use of the BM25 weighting scheme (Robertson and Zaragoza 2009) instead of the traditional term frequency-inverse document frequency (TF-IDF). BM25 (also called Okapi BM25) is a version of TF-IDF with some modifications made to each of the two components (term frequency and inverse document frequency) that, most importantly, allow it to take into account the length of the document. The classical TF-IDF formula is

$$TF - IDF = tf * log(\frac{N}{df})$$

where tf is the number of terms in the document, N is the number of documents in the corpus, and df the number of documents that contain the term. On the other hand, the formula for BM25 is the following:

$$BM25 = \frac{tf}{tf + k_1 * (1 - b + b * \frac{dl}{\overline{dl} - avg_{dl}})} * log(\frac{N - df + 0.5}{df + 0.5})$$

where k_1 is a free parameter for tuning the asymptotic maximum of the term frequency component of the equation, dl is a document length, avg_{dl} an average length of a document in the corpus, and b a free parameter for fine-tuning the document length normalization part of the equation. While Bestgen (2017) showed in his experiments that the choice of the weighting scheme does impact the performance of the classifier, the general employment of different weighting schemes by the best performing systems in past shared tasks (Zampieri *et al.* 2017) suggests that feature weighting in general is positively correlated with gains in classification performance.

A very similar SVM-based system but with simpler features (just word unigrams, bigrams and, character three- to five-grams) was used by the winners of the PAN 2017 competition Basile *et al.* (2017). The authors of the paper also discovered that adding more complex features into the model actually negatively affected its performance. An SVM ensemble with almost identical features (word unigrams and character one- to six-grams) was also used by the winners of the VarDial 2016 ADI task Malmasi *et al.* (2016*a*). An even more minimalistic SVM-based approach was proposed by the winners of the VarDial 2016 DSL competition (Çöltekin and Rama 2016), who used only character three- to seven-grams as features. The authors also report on the failed attempt to build two neural networks capable of beating the results achieved by the SVM, first one being the FastText model proposed by Joulin *et al.* (2016) and the second one a hierarchical model based on character and word embeddings. Another attempt of tackling the task with a neural approach was reported by Criscuolo and Aluisio (2017). They ranked ninth with a hybrid configuration composed of a word-level multi-layer-perceptron model and a character-level Naive Bayes model. They also experimented with a word-level convolutional neural network (CNN), which performed slightly worse than their hybrid classifier.

There have also been some quite successful attempts of tackling the language variety prediction with neural networks. Miura *et al.* (2017) ranked fourth in the PAN 2017 shared task by using a system consisting of a recurrent neural network layer, a CNN layer, and an attention mechanism. In a set of VarDial 2018 evaluation campaign tasks, Ali tackled the tasks of distinguishing between four different Swiss German dialects (Ali 2018*a*), five Arabic dialects (Ali 2018*b*), and five closely related languages from the Indo-Aryan language family (Ali 2018*c*), ranking second in the first and second task and fourth in the third task, respectively. The system is based on character-level CNNs and recurrent networks. The one-hot encoded input sequence of characters enters the network through the recurrent GRU layer used as an embedding layer. Next is the convolutional layer with different filter sizes, ranging from two to seven, which is followed by a batch normalization, max-pooling, dropout, and finally a softmax layer used for calculating the probability distribution over the labels.

While neural networks were not a frequent choice in VarDial DSL 2017 (Zampieri *et al.* 2017), in the VarDial DSL 2016 shared task (Malmasi *et al.* 2016*b*) three teams used some form of CNN. Belinkov and Glass (2016) used a character-level CNN and ranked sixth out of seven teams, achieving more than six percentage points lower accuracy than the winning system. A somewhat more sophisticated system was employed by Bjerva (2016), who combined CNN with recurrent units, developing a so-called residual network that takes as input sentences represented at a byte level. He ranked fifth in the competition. A third team called *Uppsala* used a word-level CNN but did not submit a report about their approach.

Two rear occasions when an SVM-based system did not win in a language variety classification shared task occurred at VarDial 2018 GDI and Indo-Aryan Language Identification (ILI) tasks, where Jauhiainen *et al.* beat the nearest competition by a large margin of four percentage points (Jauhiainen, Jauhiainen, and Lindén (2018a)) and more than five percentage points (Jauhiainen, Jauhiainen, and Lindén (2018b)), respectively. Their Helsinki language identification (HeLI) method with adaptive language modeling was in both cases calculated on character four-grams. The HeLI system was, however, outperformed by a margin of almost five percentage points at the VarDial 2018 Discriminating between Dutch and Flemish in Subtitles task by an SVM-based system proposed by Çöltekin, Rama, and Blaschke (2018).

3. System architecture

Research presented in Section 2 indicates that using character-level CNNs might be the most promising neural approach to the task of DSL. CNNs are able to identify important parts of a text sequence by employing a max-over-time pooling operation (Collobert *et al.* 2011), which keeps only the character sequences with the highest predictive power in the text. These sequences of predefined lengths resemble character *n*-grams, which were used in nearly every winning approach in the past shared task, but the CNN approach also has the advantage over the traditional BON approaches that it preserves the order in which these text areas with high predictive power appear in the text.

On the other hand, its main disadvantage could be the lack of an effective weighting scheme that would be capable of determining how specific these character sequences are for every input document. The data are fed into a neural classifier in small batches; therefore, it is impossible for it to obtain a somewhat global view on the data and its structure, which is encoded in the more traditional TF-IDF (or BM25) weighted input matrix. Another intuition that might explain the usefulness of weighting schemes for the specific task of language variety classification is related to named entities, for which it was shown in the past shared tasks that they in many cases reflect the origin of the text (Zampieri *et al.* 2015). The hypothesis is that these entities are quite rare and somewhat document specific and are therefore given large weights by different weighting schemes, encouraging the classifier to pay attention to them. The importance of choosing an effective weighting scheme on the task of DSL is also emphasized in the research by Bestgen

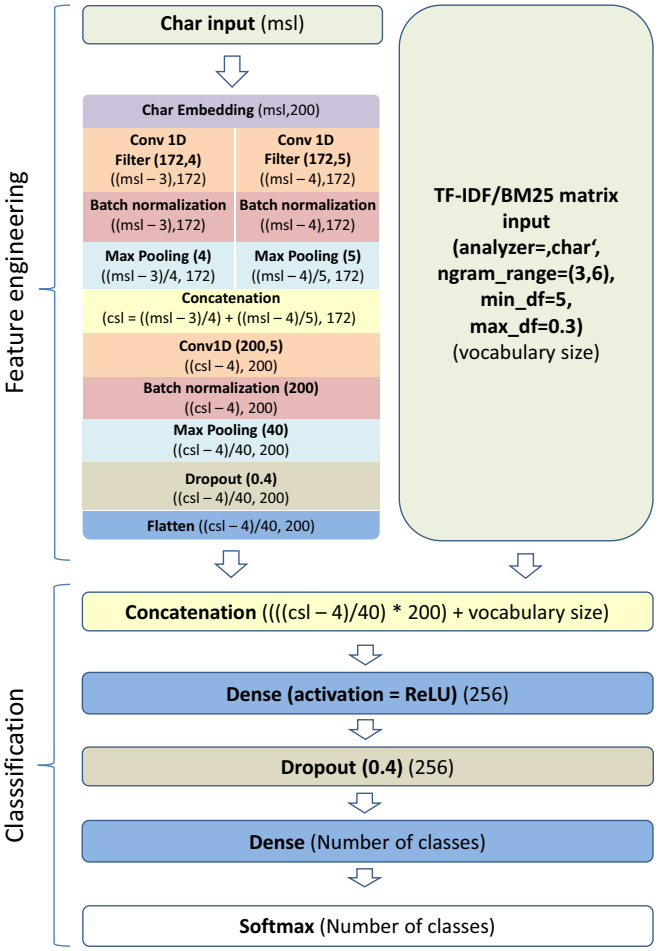


Figure 1. System architecture: layer names and input parameters are written in bold, layer output sizes are written in normal text, *msl* stands for maximum sequence length, and *csl* stands for concatenated sequence length.

(2017), the winner of the VarDial 2017 DSL task, who managed to gain some performance boost by replacing the TF-IDF weighting scheme with BM25.

Our architecture (visualized in Figure 1) builds on these findings from the literature and is in its essence an effective hybrid between a traditional feature engineering approach, which relies on different kinds of BON features, and a newer neural feature engineering approach to text classification. This combination of two distinct text classification architectures is capable of leveraging character-level and more global document/corpus-level information and achieving synergy between these two data flows. The main idea is to improve on standard CNN approaches by adding an additional input to the network that would overcome the lack of an effective weighting scheme. Therefore, the text is fed to the network in the form of two distinct inputs (as presented in Figure 1):

- *Char input*: Every document is converted into a numeric character sequence (every character is represented by a distinct integer) of length corresponding to the number of characters in the longest document in the train set (zero value padding is added after the document character sequence and truncating is also performed at the end of the sequence if the document in the validation or test set is too long).
- *TF-IDF/BM25 matrix*: We explore the effect of two distinct weighting schemes on the performance of the classifier; therefore, input data set is converted into a matrix of either TF-IDF

or BM25 weighted features with a *TfidfVectorizer* from ScikitLearn (Pedregosa *et al.* 2011) or our own implementation of the *BM25Vectorizer*. The matrix is calculated on character *n*-grams of sizes three, four, five, and six with a minimum document frequency of five and appearing in at most 30% of the documents in the train set. Sublinear term frequency scaling is applied in the term frequency calculation when *TfidfVectorizer* is used and for BM25 weighting parameters b and k_1 are set to 0.75 and 1.2, respectively, same as in Bestgen (2017).

The architecture for processing *Char input* is a relatively shallow character-level CNN with randomly initialized embeddings of size $msl \times 200$, where *msl* stands for *maximum sequence length*. Assuming that w is a convolutional filter, b is a bias, and f a nonlinear function (a rectified linear unit (*ReLU*) in our case), a distinct character *n*-gram feature c_i is produced for every possible window of h characters $x_{i:i+h-1}$ in the document according to the convolutional equation:

$$c_i = f(w \cdot x_{i:i+h-1} + b)$$

In the first step, we employ two parallel convolutional layers (one having a window of size four and the other of size five), each of them having 172 convolutional filters. These layers return two feature maps of size $(msl - ws + 1) \times 172$, where *ws* is the window size. Batch normalization and max-over-time pooling operations (Collobert *et al.* 2011) are applied on both feature maps in order to filter out features with low predictive power. These operations produce two matrices of size $(msl - ws + 1)/mws \times 172$, where sizes of max-pooling windows (*mws*) correspond to convolution window sizes. Output matrices are concatenated and the resulting matrix is fed into a second convolutional layer with 200 convolutional filters and window size five. Batch normalization and max-over-time pooling are applied again and after that we conduct a dropout operation on the output of the layer, in which 40% of input units are dropped in order to reduce overfitting. Finally, the resulting output is flattened (changed from a two-dimensional to a one-dimensional vector) and passed to a *Concatenation* layer, where it is concatenated with the input *TF-IDF/BM25 matrix*. The resulting concatenation is passed on to a fully connected layer (*Dense*) with a *ReLU* activation layer and dropout is conducted again, this time on the concatenated vectors. A final step is passing the resulting vectors to a dense layer with a *Softmax* activation, responsible for producing the final probability distribution over language variety classes.

4. Experimental setup

This section describes the data sets and the methodology used in our experiments.

4.1 Data

All experiments were conducted on three corpora described in Table 2:

- **DSLCC v4.0 (Tan *et al.* 2014)**[f]: the corpus used in the VarDial 2017 DSL shared task. The corpus contains 294,000 short excerpts of news texts divided into 6 distinct language groups (Slavic, Indonesian and Malay, Portuguese, Spanish, French, and Farsi) and covering 14 language varieties in total: Bosnian, Croatian and Serbian; Malay and Indonesian; Persian and Dari; Canadian and Hexagonal French; Brazilian and European Portuguese; Argentine, Peninsular, and Peruvian Spanish. Each language contains 20,000 documents for training (out of which 2000 are to be used as a validation set) and 1000 for testing.
- **ADIC (Ali *et al.* 2015)**[g]: the corpus used in the VarDial 2016 ADI shared task. It contains transcribed speech in Modern Standard Arabic, Egyptian, Gulf, Levantine, and North African

[f]The corpus is publicly available at http://ttg.uni-saarland.de/resources/DSLCC/

[g]The corpus is publicly available at http://alt.qcri.org/resources/ArabicDialectIDCorpus/varDial_DSL_shared_task_2016_subtask2/

Table 2. DSLCC v4.0, ADIC and GDIC corpora

DSLCC v4.0					
Language/Variety	Class	Train inst.	Train tokens	Test inst.	Test tokens
Bosnian	bs	20,000	716,537	1000	35,756
Croatian	hr	20,000	845,639	1000	42,774
Serbian	sr	20,000	777,363	1000	39,003
Indonesian	id	20,000	800,639	1000	39,954
Malay	my	20,000	591,246	1000	29,028
Brazilian Portuguese	pt-BR	20,000	907,657	1000	45,715
European Portuguese	pt-PT	20,000	832,664	1000	41,689
Argentine Spanish	es-AR	20,000	939,425	1000	42,392
Castilian Spanish	es-ES	20,000	1,000,235	1000	50,134
Peruvian Spanish	es-PE	20,000	569,587	1000	28,097
Canadian French	fr-CA	20,000	712,467	1000	36,121
Hexagonal French	fr-FR	20,000	871,026	1000	44,076
Persian	fa-IR	20,000	824,640	1000	41,900
Dari	fa-AF	20,000	601,025	1000	30,121
Total		280.000	8,639,459	14,000	546,790
ADIC					
Egyptian	EGY	1578	85,000	315	13,000
Gulf	GLF	1672	65,000	256	14,000
Levantine	LAV	1758	66,000	344	14,000
Modern Standard	MSA	999	49,000	274	14,000
North African	NOR	1612	52,000	351	12,000
Total		7619	317,000	1540	67,000
GDIC					
Bern	BE	4956	35,962	1191	12,013
Basel	BS	4921	36,965	1200	9802
Lucerne	LU	4593	38,328	1186	11,372
Zurich	ZH	4834	36,919	1175	9610
Total		19,304	148,174	4,752	42,797

dialects. Speech excerpts were taken from a multi-dialectical corpus containing broadcast, debate and discussion programs from Al Jazeera. Altogether 7619 documents were used for training (out of which 10% were used for validation) and 1540 documents for testing.

- **GDIC (Samardzic, Scherrer, and Glaser (2016))**: the corpus used in the VarDial 2018 GDI shared task. Texts were extracted from the ArchiMob corpus of Spoken Swiss German[h], which contains 34 oral interviews with people speaking Bern, Basel, Lucerne, and Zurich Swiss German dialects. A total of 19,304 documents were used for training (out of which 10% were used for validation) and 4752 for testing.

[h]The ArchiMob corpus is publicly available at https://www.spur.uzh.ch/en/departments/research/textgroup/ArchiMob.html

4.2 Methodology

For experiments in the DSLCC v4.0 we chose to use a two-step approach, as first proposed by Goutte, Léger, and Carpuat (2014):

(1) The general classifier is trained to identify the language group for every specific document. For this step, the input TF-IDF/BM25 matrix is calculated only on the word bound character *n*-grams[i] of sizes three, four, and five with a minimum document frequency of five and appearing in at most 30% of the documents in the train set. This configuration produces a TF-IDF/BM25 matrix of smaller size than if the configuration for the TF-IDF/BM25 matrix, described in Section 3, was used. This size reduction was chosen because distinguishing between different language groups is not a difficult problem, therefore, this parameter reduction does not influence performance but it reduces the execution time.
(2) We train six different classification models, one for each language group. After being classified as belonging to a specific language group by the general classifier in Step 1, the documents are assigned to the appropriate classifier for predicting the final language variety.

Since NLP tools and resources such as POS taggers, pretrained word embeddings, word dictionaries, and tokenizers might not exist for some underresourced languages, we also believe that an architecture which does not require language-specific resources and tools, apart from the training corpus, might be more useful and easier to use in real-life applications. For this reason, our system does not require any additional resources and the conducted preprocessing procedure is light[j].

We show (see Section 5) that the proposed architecture is generic enough to outperform the winning approach of VarDial 2017 on all of the language groups without any language-group-specific parameter or architecture tweaking. In contrast, most of the approaches of the VarDial 2017 DSL shared task resorted to language-group-specific optimization, as getting even the slightest possible performance boost by employing this tactic was important due to the competitive nature of shared tasks.

For the experiments on the smaller ADIC and GDIC data sets, we use the same hyperparameter configuration and TD-IDF/BM25 features as for the six classification models for specific language groups in the DSLCC v4.0 corpus because we want to explore the relation between model performance and data set size. The hypothesis is that the performance of traditional SVM approaches would be less affected by smaller data set size than neural approaches.

We conducted an extensive grid search on the DSLCC v4.0 in order to find the best hyperparameters for the model. All combinations of the following hyperparameter values were tested before choosing the best combination, which is written in bold in the list below and presented in Section 3:

- Learning rates: 0.001, **0.0008**, 0.0006, 0.0004, 0.0002
- Number of parallel convolutions with different filter sizes: [3] [4], [3,4], **[4,5]**, [5,6], [6,7], [3,4,5], [4,5,6], [5,6,7], [3,4,5,6], [4,5,6,7], [3,4,5,6,7]
- Character embedding sizes: 100, **200**, 400
- Dense layer sizes: 128, **256**, 512
- Dropout values: 0.2, 0.3, **0.4**, 0.5
- Number of convolutional filters in the first convolution step: 156, **172**, 200
- Number of convolutional filters in the second convolution step: 156, 172, **200**

[i]Word-bound character *n*-grams are made only from text inside word boundaries, for example, a sequence *this is great* would produce a word-bound character 4-gram sequence *this, is__, grea, reat*, in which _ stands for empty space character.

[j]We only replace all email addresses in the text with *EMAIL* tokens and all URLs with *HTTPURL* tokens by employing regular expressions. Even if this might not be relevant to all of the corpora, we keep the preprocessing unchanged for all the settings.

- Size of a max-pooling window in the second convolution step: 10, 20, **40**, 60
- BON n sizes: [3] [4] [3,4], [4,5], [5,6], [6,7], [3,4,5], [4,5,6], [5,6,7], **[3,4,5,6]**, [4,5,6,7], [3,4,5,6,7]
- Minimum document frequency of an n-gram in the TF-IDF/BM25 matrix: [2], **[5]**, [10]
- BM25 b parameter: 0.5, **0.75**, 1.0
- BM25 k_1 parameter: 1.0, **1.2**, 1.4

The hyperparameters, which influenced the performance of the network the most, were the learning rate, CNN filter sizes, size of the max-pooling window, BON n size, and a minimum document frequency of n-grams. Too many parallel convolutions, small sizes of the max-pooling window, and low minimum document frequency of n-grams showed tendency toward overfitting, especially when used together in combination. In general, we noticed quite a strong tendency toward overfitting no matter the hyperparameter combination, which could be to some extent the consequence of feeding a high-dimensional TF-IDF/BM25 matrix to the network, which greatly increases the number of network parameters. We noticed that a combination of a relatively small learning rate and a large dropout worked best to counter this tendency.

Another thing we noticed is that using exactly the same configurations of convolutional filter sizes and n-gram sizes negatively affected the performance, which was slightly improved when the configurations did not completely overlap. The hypothesis is that synergy between two data flows is less effective if the information in these two data flows is too similar. The validation set results did however show that configurations containing 4- and 5-grams and filter sizes of 4 and 5 in general worked better than other configurations for DSLCC v4.0 classification; therefore, these configurations were used in both data flows despite the overlap.

We use the Python Keras library (Chollet 2015) for the implementation of the system. For optimization, we use an Adam optimizer (Kingma and Ba 2015) with a learning rate of 0.0008. For each language variety in the DSLCC v4.0, the model is trained on the train set for 20 epochs and tested on the validation set after every epoch. The models trained on the ADIC and GDIC data sets are trained for 80 epochs due to longer convergence time on less data. The model with the best performance on the validation set is chosen for the test set predictions.

5. Results

First we present results on the DSLCC v4.0, which is (as it is the largest and covers the largest number of language varieties) the main focus of this study, then we present results on ADIC and GDIC and finally, we present findings of the error analysis conducted on the misclassified Slavic documents of the DSLCC v4.0 corpus.

5.1 Results on the DSLCC v4.0

Table 3 presents the results achieved by our neural classifier in comparison to the winner of the VarDial 2017 DSL shared task (Bestgen 2017) in terms of weighted F1, micro F1, macro F1, and accuracy measures.

The first step of the two-step classification approach, distinguishing between different language groups (*All-language groups (TF-IDF)* and *All-language groups (BM25)* rows in Table 3), proved trivial for the system, which achieved almost perfect weighted F1 score and misclassified only 27 documents out of 14,000 in the test set when TF-IDF weighting scheme was used and 29 documents when BM25 weighting scheme was used. If we look at the confusion matrices for language group classification (Figures 2 and 3), both models had most difficulties distinguishing between Spanish and Portuguese language groups. Ten Spanish texts were misclassified as Portuguese but on the other hand, only one Portuguese document was misclassified as Spanish when TF-IDF weighting scheme was used. With BM25 weights, the classifier misclassified nine

Table 3. Results of the proposed language variety classifier on the DSLCC v4.0 for different language groups, as well as for the discrimination between language groups (All-language groups). Also the results for all language varieties (All-language varieties) are provided, for which a comparison with the official VarDial 2017 winners is made. Results for both weighting schemes, TF-IDF and BM25, are reported separately

Language group (weighting)	F1 (weighted)	F1 (micro)	F1 (macro)	Accuracy
All-language groups (TF-IDF)	**0.9981**	**0.9981**	**0.9980**	**0.9981**
All-language groups (BM25)	0.9979	0.9979	0.9980	0.9980
Spanish (TF-IDF)	**0.9136**	**0.9140**	**0.9136**	**0.9140**
Spanish (BM25)	0.9042	0.9047	0.9042	0.9047
Slavic (TF-IDF)	0.8645	0.8650	0.8645	0.8650
Slavic (BM25)	**0.8752**	**0.8753**	**0.8752**	**0.8753**
Farsi (TF-IDF)	0.9685	0.9685	0.9685	0.9685
Farsi (BM25)	**0.9690**	**0.9690**	**0.9690**	**0.9690**
French (TF-IDF)	**0.9570**	**0.9570**	**0.9570**	**0.9570**
French (BM25)	0.9545	0.9545	0.9545	0.9545
Malay and Indonesian (TF-IDF)	0.9855	0.9855	0.9855	0.9855
Malay and Indonesian (BM25)	**0.9860**	**0.9860**	**0.9860**	**0.9860**
Portuguese (TF-IDF)	**0.9480**	**0.9480**	**0.9480**	**0.9480**
Portuguese (BM25)	0.9460	0.9460	0.9460	0.9460
All-language varieties (TF-IDF)	**0.9310**	**0.9312**	**0.9310**	**0.9312**
All-language varieties (BM25)	0.9304	0.9305	0.9304	0.9305
VarDial 2017 winner Bestgen (2017)	0.9271	0.9274	0.9271	0.9274

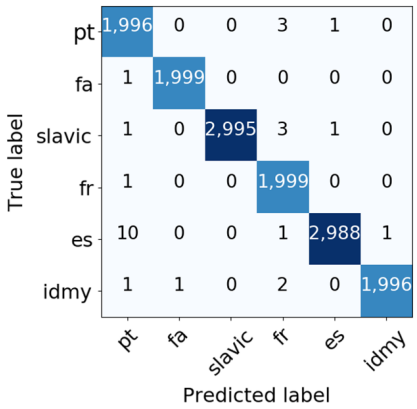


Figure 2. Confusion matrix for language group classification (TF-IDF weighting scheme).

Spanish documents as Portuguese and four Portuguese documents as Spanish. The analysis also reveals some surprising mistakes, such as that three Slavic documents and two documents from the Indonesian and Malay language group were misclassified as French with TF-IDF weighting and four documents from the Indonesian and Malay language group, three Spanish, and three French documents were classified as Slavic with BM25 weighting. A closer inspection of misclassified documents also reveals that these documents are in general much shorter (average word length is 9.74 and 10.17 when TF-IDF and BM25 are used respectively) than an average document in the Slavic sub-corpus (39.06 words long) and very likely contain some misleading named entities (e.g., a Slavic document, which was misclassifed as Spanish when TF-IDF weighting was used, contains the following text: *Caffe - Pizzeria ""BELLA DONNA"" u DOC-u*).

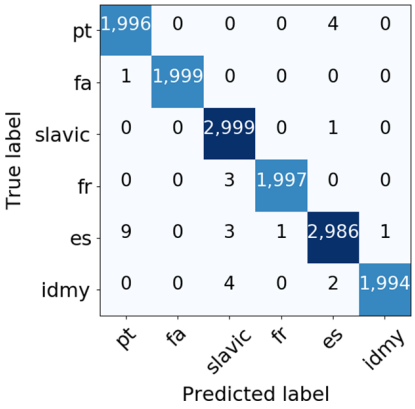


Figure 3. Confusion matrix for language group classification (BM25 weighting scheme).

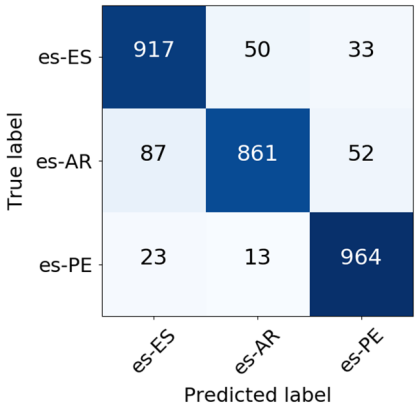


Figure 4. Confusion matrix for Spanish language varieties classification (TF-IDF weighting scheme).

Results for the second step of the two-step classification approach indicate that the difficulty of distinguishing language varieties within different language groups varies. The system had most difficulties with distinguishing between different Slavic languages, where it achieved by far the worst results with an weighted F1 of 0.8645 when TF-IDF weighting scheme was employed and about one percentage point better results when BM25 weighting was used. The second most difficult were Spanish varieties. We should point out that this comes as no surprise, since Slavic and Spanish languages groups were the only two groups that contained three varieties, while the other groups in DSLCC v4.0 contained two varieties. The system had least problems with distinguishing between Malay and Indonesian languages.

When it comes to comparing two weighting schemes, there is no clear overall winner. The biggest differences in performance are on Spanish varieties, where TF-IDF weighting outperforms BM25 by about one percentage point according to every measure, and on Slavic varieties, where BM25 weighting outperforms TF-IDF by a very similar margin. The differences on other varieties are smaller, ranging from 0.005 on Farsi and Malay and Indonesian varieties to 0.020 on Portuguese varieties.

Confusion matrices for specific language varieties enable a more thorough analysis of the results. For Spanish varieties (Figures 4 and 5), the system had most problems distinguishing between Argentine and Castilian Spanish. The second most common mistake no matter the weighting scheme was classifying Argentine Spanish as the Peruvian variety of Spanish. On the other hand, Peruvian Spanish was the easiest to classify by the system, with altogether only 36 (TF-IDF weighting) and 37 (BM25 weighting) misclassified instances.

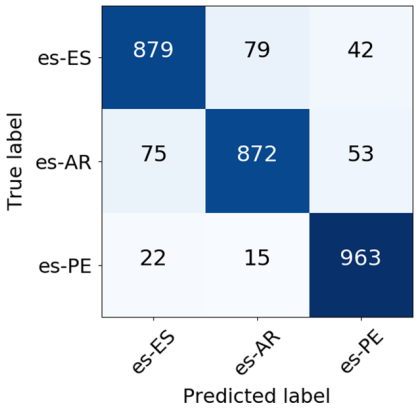


Figure 5. Confusion matrix for Spanish language varieties classification (BM25 weighting scheme).

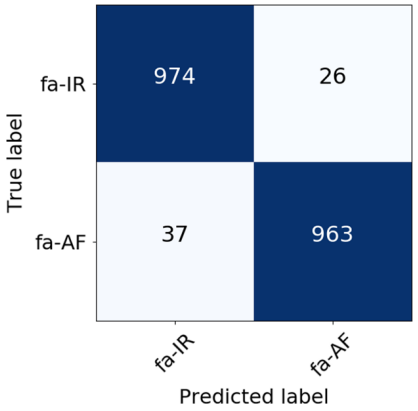


Figure 6. Confusion matrix for Farsi language varieties classification (TF-IDF weighting scheme).

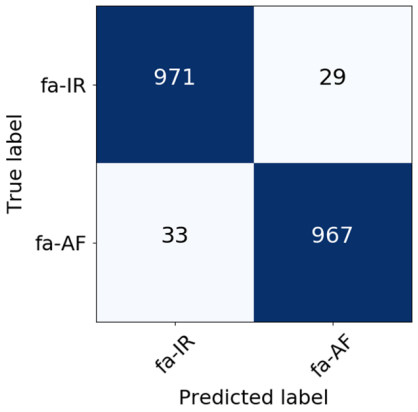


Figure 7. Confusion matrix for Farsi language varieties classification (BM25 weighting scheme).

The system performed well for all binary predictions (Figures 6 and 7, Figures 8 and 9, Figures 10 and 11, Figures 12 and 13) and the difference in performance between two weighting schemes are small. Out of these confusion matrices, the most unbalanced with regard to false predictions is the confusion matrix for Indonesian and Malay variety (Figure 10), where twice as many Indonesian documents were classified as Malay than the other way around when TF-IDF weighting was used. Although, as mentioned before, distinguishing between Indonesian and Malay was the least difficult task for the classifier and altogether only 29 and 28 instances were misclassified when TF-IDF and BM25 weighting were used, respectively.

For Slavic languages (Figures 14 and 15), the hardest problem for the system was distinguishing between Croatian and Bosnian, with 113 Bosnian documents being classified as Croatian and 112 Croatian documents being classified as Bosnian when TF-IDF weighting was used and with 113

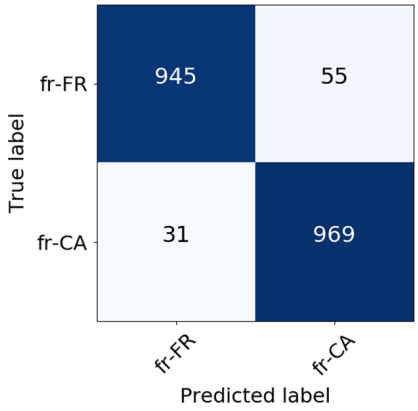


Figure 8. Confusion matrix for French language varieties classification (TF-IDF weighting scheme).

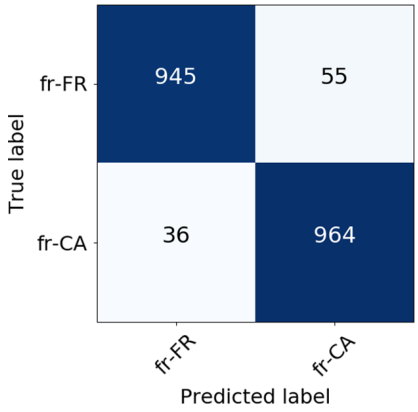


Figure 9. Confusion matrix for French language varieties classification (BM25 weighting scheme).

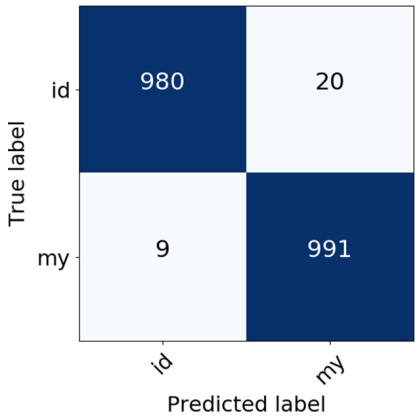


Figure 10. Confusion matrix for Indonesian and Malay variety classification (TF-IDF weighting scheme).

Bosnian documents being classified as Croatian and 99 Croatian documents being classified as Bosnian when BM25 weighting was employed. Distinguishing between Bosnian and Serbian was also not trivial for the classifier no matter the weighting scheme, with 94 Bosnian documents being misclassified as Serbian and 66 Serbian documents misclassified as Bosnian when TF-IDF weighting scheme was deployed and 73 Bosnian documents being misclassified as Serbian and vice versa when BM25 weighting was used. On the other hand, distinguishing between Serbian and Croatian is a much easier problem, with altogether only 20 (TF-IDF weighting) and 16 (BM25 weighting) documents being misclassified.

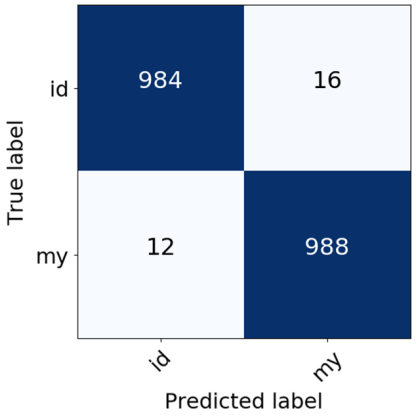


Figure 11. Confusion matrix for Indonesian and Malay variety classification (BM25 weighting scheme).

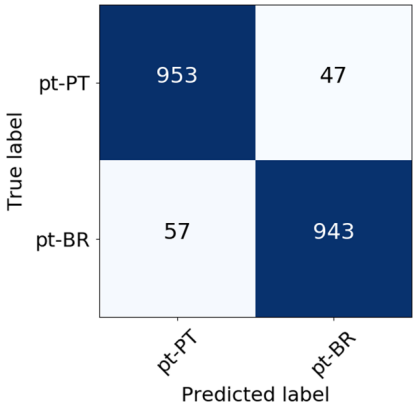


Figure 12. Confusion matrix for Portuguese language varieties classification (TF-IDF weighting scheme).

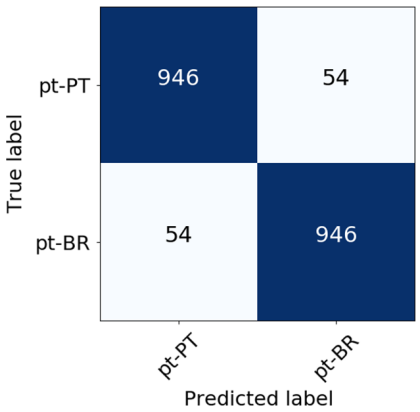


Figure 13. Confusion matrix for Portuguese language varieties classification (BM25 weighting scheme).

Overall (rows *All-language varieties (TF-IDF)* and *All-language varieties (BM25)* in Table 3), the neural network outperforms the SVM-based approach used by the winners of the shared task by about 0.4 percentage points according to all measures when TF-IDF weighting scheme is used. BM25 weighting performs slightly worse but still outperforms state of the art by about 0.35 percentage points margin. Our results therefore differ from the study conducted by Bestgen (2017), the winner of the shared task, where he reported improvement in performance for all but one language group when TF-IDF weighting is replaced by BM25. It should, however, be noted that these improvements were only reported on the validation set and no comparison between weighting schemes was done on the official test set.

There were no available reported results for individual language groups on the official test set, therefore we provide a comparison with the VarDial 2017 DSL winning team on the validation set,

Table 4. Accuracy comparison of our system to the VarDial 2017 DSL winners on validation sets

Language group (weighting)	Our system	VarDial 2017 winner	Improvement (%)
Spanish (TF-IDF)	**0.9180**	0.8970	2.10
Spanish (BM25)	**0.9202**	0.9030	1.72
Slavic (TF-IDF)	**0.8663**	0.8445	2.18
Slavic (BM25)	**0.8670**	0.8506	1.64
Farsi (TF-IDF)	**0.9685**	0.9598	0.87
Farsi (BM25)	**0.9720**	0.9632	0.88
French (TF-IDF)	**0.9588**	0.9396	1.92
French (BM25)	**0.9590**	0.9472	1.18
Malay and Indonesian (TF-IDF)	**0.9863**	0.9835	0.28
Malay and Indonesian (BM25)	**0.9875**	0.9827	0.48
Portuguese (TF-IDF)	**0.9440**	0.9299	1.41
Portuguese (BM25)	**0.9428**	0.9355	0.73

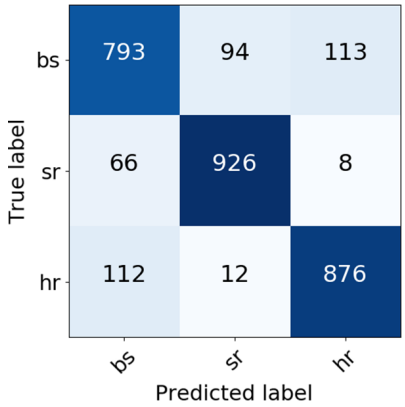


Figure 14. Confusion matrix for Slavic language varieties classification (TF-IDF weighting scheme).

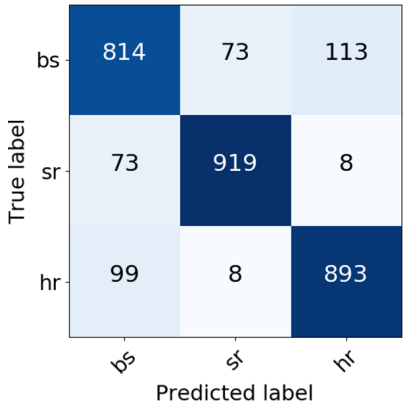


Figure 15. Confusion matrix for Slavic language varieties classification (BM25 weighting scheme).

as the author (Bestgen 2017) reports them when presenting the benefits of the weighting scheme BM25 (in their Table 3 on p. 119). Note, however, that the results report on a slightly simplified system, as for the weighting scheme comparison, the author used only character *n*-grams features. Comparison results are presented in Table 4.

Table 5. Results of the proposed language variety classifier on the ADIC and GDIC. Results for both weighting schemes, TF-IDF and BM25, are reported separately

Language group (weighting)	F1 (weighted)	F1 (micro)	F1 (macro)	Accuracy
ADIC (TF-IDF)	**0.5152**	0.5123	0.5147	**0.5123**
ADIC (BM25)	0.5090	0.5097	0.5067	0.5097
VarDial ADI 2016 winner Malmasi *et al.* (2016*a*)	0.5132	/	/	0.5117
GDIC (TF-IDF)	0.6281	0.6294	0.6280	0.6294
GDIC (BM25)	0.6289	0.6311	0.6289	0.6311
VarDial GDI 2018 winner Jauhiainen *et al.* (2018*a*)	/	/	**0.6860**	/

Our system performs better than the simplified version of the VarDial 2017 DSL shared task winning system on all language groups. When TF-ID weighting is used by both systems, the differences vary from around two percentage points on Spanish, Slavic, and French language groups, to about 1.5 percentage point difference on the Portuguese language group, and finally, to only 0.28 percentage point difference on Malay and Indonesian, which are the easiest languages to distinguish for both of the classifiers. When BM25 weighting scheme is used, the differences are smaller, ranging from about 1.5 percentage point on Spanish and Slavic to about 0.5 percentage point on Malay and Indonesian.

Interestingly, when it comes to comparing both weighting schemes only on validation sets, the influence on the performance of our system when BM25 weighting is used is quite consistent with the influence reported by Bestgen (2017). By using BM25 weighting, the performance is improved on five out of six language groups, same as in Bestgen (2017), although the language groups are not the same: in Bestgen (2017) performance is not improved on the Malay and Indonesian language group while we report no improvement on Portuguese. However, these improvements at least in our case do not translate well to performance improvements on the official test set.

5.2 Results on ADIC and GDIC

Table 5 presents the results achieved by our neural classifier on the ADIC and GDIC corpora in comparison to the winners of the VarDial ADI 2016 and VarDial GDI 2018 shared tasks. The system manages to improve on the state of the art on the ADIC by a small margin of about 0.2 percentage point according to the weighted F1 score when TF-IDF weighting is used, even though the ADIC contains more than 10 times less documents per class than the language varieties in the DSLCC v4.0. By using BM25 weighting, the performance of the classifier is about 0.6 and 0.2 percentage points worse in terms of accuracy and weighted F1 score. On the other hand, the results on the GDIC are almost six percentage points lower than the current state-of-the-art HeLI method (Jauhiainen *et al.* 2018*a*) in terms of macro F1 score. Our system also performed worse than the SVM-based system proposed by Çöltekin *et al.* (2018) and a recurrent neural network proposed by Ali (2018a), which achieved macro F1 scores of 0.646 and 0.645, respectively. We can also observe that BM25 weighting slightly improves the performance according to all the criteria. Results on ADIC and GDIC corpora are somewhat in line with the initial hypothesis that neural approaches are more affected by a small data set size than more traditional SVM approaches. Previous SVM-based state of the art on the ADIC corpora is outperformed by a smaller margin than the DSLCC v4.0 state of the art and the proposed system performs worse than the second ranked SVM system (2018) on the GDIC corpus.

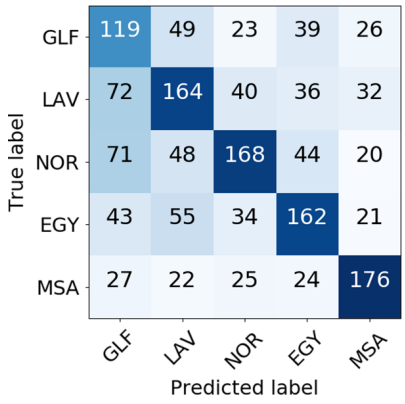


Figure 16. Confusion matrix for Arabic language varieties classification (TF-IDF weighting scheme).

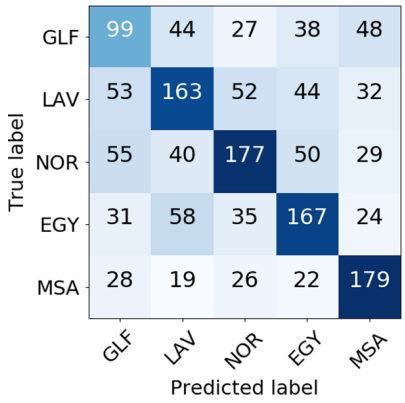


Figure 17. Confusion matrix for Arabic language varieties classification (BM25 weighting scheme).

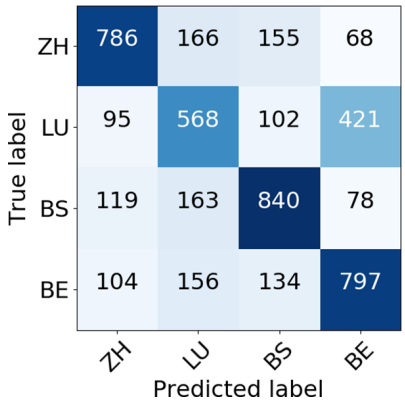


Figure 18. Confusion matrix for German language varieties classification (TF-IDF weighting scheme).

Confusion matrices for the ADIC (Figures 16 and 17) show that the Modern Standard Arabic is the easiest to classify no matter the weighting scheme. We can also see that if BM25 weighting is used, the classifier struggles much more with the Gulf dialect, correctly classifying only 99 out of 256 instances, than if TF-IDF weighting is used, in which case it correctly classifies 119 instances.

Confusion matrices for the GDIC (Figures 18 and 19) show that the choice of the weighting scheme does not have as big of an influence on the performance of the classifier as in the case of ADIC. No matter the weighting scheme, by far the most common mistake was misclassifying the Lucerne dialect as a Bern dialect. Interestingly, the opposite mistake of misclassifying Bern dialect

Table 6. Results of the error analysis on 405 misclassified Slavic documents

Group	Num. doc.	Prop. of doc.	Avg. doc. length
No named entities	144	0.36	26.94
Misleading named entities	70	0.17	40.96
Clarifying named entities	41	0.10	34.96
Unrelated named entities	150	0.37	33.17
All misclassified	405	1.00	32.48

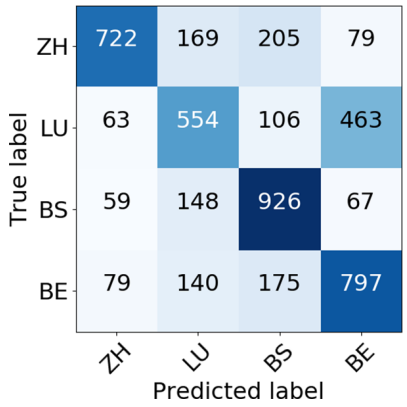


Figure 19. Confusion matrix for German language varieties classification (BM25 weighting scheme).

as Lucerne dialect is much rarer, which might be connected to some extent to the fact that the train set contains 328 more documents for the Bern dialect than for the Lucerne dialect.

5.3 Error analysis

We conducted a manual error analysis on the misclassified Slavic documents[k] in order to get a clearer picture about what kind of documents are the hardest to classify. Misclassified documents were manually grouped into four classes according to the number and type of named entities found in the document:

- **No named entities**: Documents without any named entities.
- **Misleading named entities**: Documents containing any named entities (e.g., names of regions, cities, public figures) originating from a country with the official language variety corresponding to one of the two possible incorrect language varieties (e.g., a document labeled as Serbian containing the word *Zagreb*, which is the capital of Croatia, would be put into this class).
- **Clarifying named entities**: Documents containing named entities originating from a country with the official language variety being the correct language variety and containing no misleading entities.
- **Unrelated named entities**: Documents containing only named entities that are not originating from any of the countries speaking target language varieties (e.g., a document containing only the named entity *Budapest* would be classified into this category).

Results of the analysis are presented in Table 6. Results show that a large portion of misclassified documents (73%) either contain no named entities (36%) or contain only unrelated named entities (37%), which might make them harder to classify, although we cannot claim that for sure, since we

[k]Error analysis was conducted on documents misclassified by the system that employed TF-IDF weighting scheme.

Table 7. Results of the ablation study. Column *CNN F1 (weighted)* presents performance of the system in terms of weighted F1 if only CNN-based features are used, column *BON F1 (weighted)* presents performance of the system if only TF-IDF-weighted BON features are used and column *All F1 (weighted)* presents the performance when these two types of features are combined

Language group	All F1 (weighted)	CNN F1 (weighted)	BON F1 (weighted)
DSLCC v4.0			
All-language groups	**0.9981**	0.9971	0.9976
Spanish	**0.9136**	0.8599	0.8863
Slavic	**0.8645**	0.8300	0.8594
Farsi	**0.9685**	0.9465	0.9610
French	**0.9570**	0.9325	0.9420
Malay and Indonesian	0.9855	0.9560	**0.9875**
Portuguese	**0.9480**	0.8994	0.9434
All-language varieties	**0.9310**	0.8935	0.9199
ADIC	0.5152	0.3971	**0.5177**
GDIC	**0.6281**	0.6059	0.6190

do not know the distribution of these classes across the entire test set. About 17% of the documents on the other hand contain misleading named entities that could influence the classifier prediction. There are also 41 documents (10%) containing only clarifying named entities that would be easily classified correctly by any human annotator with some basic background knowledge about Serbia, Bosnia, and Croatia. This suggests that there is still some room for improvement for the developed classifier.

Another finding is that misclassified documents are in average shorter (32.48 words long) than an average document from a Slavic language group (39.18 words long), suggesting that shorter documents are harder to classify by the classifier due to less available information. We can also see that the only group containing documents with similar length as the whole test set are documents containing misleading named entities (40.96 words long), which suggests that the classifier does somewhat rely on named entities during the prediction process.

6. Ablation study

The main novelty of our approach is the combination of weighted BON features with CNN-generated character features in the neural architecture. We carried out an ablation study in order to determine the contribution of these two types of features in the overall performance. To measure the contribution of weighted BON features, we removed the part of the system that deals with the convolutional processing of the character sequence input (the left side of the feature engineering part sketched in Figure 1). On the other hand, we removed the TF-IDF/BM25 matrix input in order to determine the contribution of the CNN-generated character features. Only TF-IDF weighting was used in the ablation study. The results of the study are presented in Table 7.

In all cases, classifier with only TF-IDF-weighted BON features (BON classifier) performs better than the classifier with only CNN-based features (CNN classifier), which also raises questions about the established deep learning paradigm that in a large majority of cases relies only on the automatically generated neural features. In DSLCC v4.0, the difference in performance is the largest in the case of Portuguese language variety classification, measuring more than four percentage points. If we ignore the language group classification, which is apparently trivial for all

Table 8. Results of the error analysis on Slavic documents misclassified by the BON classifier and correctly classified by the CNN classifier and on Slavic documents misclassifed by the CNN classifier and correctly classified by the BON classifier

Group	Num. doc.	Prop. of doc.	Avg. doc. length
BON misclassifed			
No named entities	57	0.31	27.14
Misleading named entities	17	0.09	38.18
Clarifying named entities	28	0.15	33.14
Unrelated named entities	83	0.45	35.04
All	185	1.00	32.61
CNN misclassifed			
No named entities	81	0.30	31.43
Misleading named entities	36	0.13	45.67
Clarifying named entities	58	0.21	35.53
Unrelated named entities	99	0.36	34.98
All	274	1.00	35.45

three versions of the system, the difference in performance is the smallest for the French language variety classification, only around one percentage point.

By combining both types of features, we manage to surpass the performance of the BON classifier on all language groups in the DSLCC v4.0 but the Malay and Indonesian pair. Here, the BON classifier beats the classifier with the combination of both types of features by a small margin of 0.2 percentage points. The synergy effect is the largest in case of Spanish language variety, where we improve the performance of the BON classifier by almost three percentage points. Overall performance of the classifier on all the languages is improved by about one percentage point in comparison to the BON classifier.

Results on smaller data sets are somewhat hard to generalize. In the case of ADIC, the performance gap between BON and CNN is almost 11 percentage points. The bad performance of the CNN classifier in this case also most likely outweighs any positive synergy effect, causing the classifier that uses a combination of both feature types to perform slightly (by about 0.3 percentage points) worse than the BON classifier (which is therefore a new state-of-the-art classifier for the ADIC data set). In the case of GDIC, the performance gap is smaller (about 1.3 percentage points) and there is some synergy effect between the two classifiers.

In order to determine what types of texts are better predicted with the BON classifier and what types of text are better predicted with the CNN classifier, we performed the same error analysis as in Section 5.3 on 185 Slavic documents, which were correctly classified by the CNN classifier and misclassified by the BON classifier, and on 274 documents which were correctly classified by the BON classifier and misclassified by the CNN classifier. Results are presented in Table 8. We can see that on average both of these documents are shorter (32.61 and 35.45 words long) than an average document in the Slavic sub-corpus (39.18 words long). Similar share of documents with no named entities was misclassified by both classifiers but there are differences in shares when it comes to other classes. Both BON and CNN classifiers performed the worst on documents containing only unrelated named entities but the share of these documents in the overall distribution of misclassified documents is much bigger for the BON classifier (0.45 vs. 0.36). On the other hand, documents containing clarifying named entities represent a smaller share in the distribution of documents misclassified by the BON classifier (0.15 vs. 0.21). These results are in accordance with the hypothesis that the BON classifier relies to a larger extent on named entities

than the CNN classifier. The share of documents with misleading named entities is the smallest in distributions for both classifiers, which was not the case in the error analysis in Section 5.3 (see Table 6), where the smallest share presented documents with only clarifying named entities. This suggests that both classifiers struggle with these documents and are in most cases misclassified by both classifiers; therefore (as this ablation study is focused on the differences between the BON and CNN classifiers), these documents were not manually analyzed.

7. Workflow for language variety classification

The AP—and larger NLP—community encourages reproducibility of results and code sharing[l]; therefore, our source code is published at `http://source.ijs.si/mmartinc/NLE_2017/`. Since AP is also a very interdisciplinary field, we also believe it is important to make our tools available to the users outside of the programming community (e.g., linguists or social scientists) with lower level of technical skills.

In our previous work (Martinc and Pollak 2018), we have already implemented a set of pretrained gender classification models into a cloud-based visual programming platform ClowdFlows (`http://clowdflows.org`) (Kranjc *et al.* 2012). These tools can be used out-of-the-box and are therefore appropriate for the less tech savy members of the AP community. The ClowdFlows platform employs a visual programming paradigm in order to simplify the representation of complex data mining procedures into visual arrangements of their building blocks. Its graphical user interface is designed to enable the users to connect processing components (i.e., widgets) into executable pipelines (i.e., workflows) on a design canvas, reducing the complexity of composition and execution of these workflows. The platform also enables online sharing of the composed workflows.

We took all our pretrained models for language variety classification (six models for six language groups and the general model for distinguishing between different language groups from the DSLCC v4.0, and German and Arabic models used for ADIC and GDIC classification) and packed them in a widget *Language Variety Classifier*. The widget takes a Pandas dataframe (McKinney 2011) containing the corpus as an input and returns a dataframe with an additional column with predicted language/language variety labels. The user needs to define the name of the column containing text documents as a parameter and choose the language group (or language parameter value *all* in order to use the general classifier) according to the input text.

Workflow in Figure 20 (available at `http://clowdflows.org/workflow/13322/`) is a ClowdFlows implementation of the two-step approach described in Section 4.2 for the language variety classification, illustrated on the DSLCC v4.0 test set. The corpus is loaded from a CSV file with two columns (one for texts and one for true labels) with the help of the *Load corpus from CSV* widget and passed on to the *Language variety classifier* widget, which predicts general language groups for all the texts. The *Filter corpus* widgets are used to split the corpus according to the predicted language group labels. Each of the slices is then fed into six different *Language variety classifier* widgets responsible for intra-language group classification. They output a Pandas dataframe with an additional column containing the predicted variety labels for each corpus slice. The corpus is reassembled with the help of the *Concatenate corpora* widget. The reassembled corpus and the six sub-corpora are then fed into seven *Calculate F1 and accuracy* widgets, which are in fact subprocess widgets[m], each of them containing a subprocess for calculating the accuracy and weighted F1 score[n] of the classification. The results of the classification are written to a table with the help of an *Evaluation results to table* widget. We have presented a repeatable and

[l]For example, this is the Github repository for the PAN shared task: `https://github.com/pan-webis-de`

[m]More information about the different types of widgets in the ClowdFlows platform is available at the ClowdFlows documentation page `https://clowdflows.readthedocs.io/en/latest/`.

[n]The results produced by the workflow vary very slightly from the results reported in Section 4 because Theano (Bergstra *et al.* 2011) is used as Keras backend in the ClowdFlows platform instead of Tensorflow (Abadi *et al.* 2016), which is used for producing the results reported in Section 4.

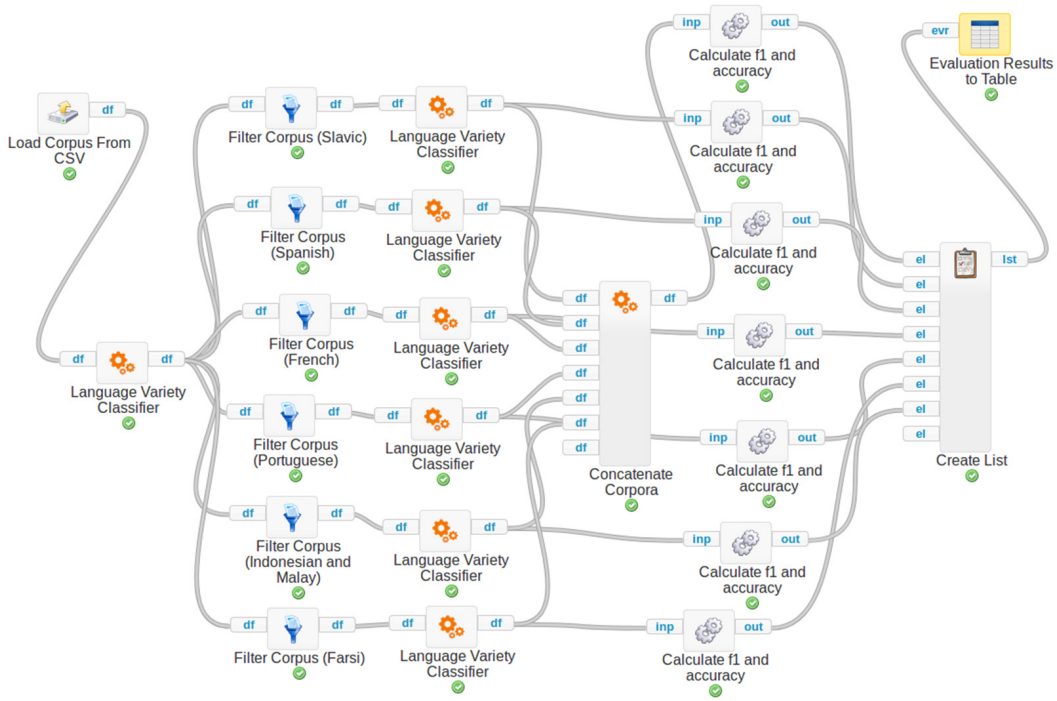


Figure 20. ClowdFlows implementation of the two-step approach for the language variety classification on the DSLCC v4.0. Workflow is publicly available at http://clowdflows.org/workflow/13322/.

transparent evaluation workflow, which can be easily tested on novel test sets, but note that the Language variety classifier widget can also be used in novel workflows, for assigning the language of unlabeled text segments. The simplest use would be to input a file with text that user wants to label in a CSV format and connect it to the two-step language classification widgets in order to obtain the labeled corpus (http://clowdflows.org/workflow/13670/).

8. Discussion and conclusions

In this paper, we present an original neural language variety classifier. The main novelty is the architecture that is capable of leveraging character-level and more global document/corpus-level information by combining weighted BON features with character-based CNN features. The system was tested on the DSLCC v4.0, ADIC and GDIC corpora, used in the VarDial shared tasks, and managed to outperform state-of-the-art approaches developed in the scope of the shared task on two (including on the benchmark DSLCC v4.0) out of three corpora. An ablation study shows that weighted BON features generally contribute more than CNN-based features. This is in accordance with the previous results in the AP shared tasks where BON-based classification systems were always the winners. On the other hand, our experiments showed that replacing TF-IDF weighting with BM25 weighting in most cases does not improve performance, which is not in accordance with the previous research (Bestgen 2017). Our system is also openly available as a workflow in the ClowdFlows platform for less tech savy members of the AP community.

The experiments on the DSLCC v4.0 have shown that building a neural architecture outperforming the popular SVM BON classification combination on the language variety task is possible, although the performance gains are not very large. With some additional language-group-specific parameter tweaking the performance could be improved, but we decided against this idea in order to preserve the generic nature of the common architecture, which is currently capable of producing state-of-the-art predictions for six different language groups.

The system also proved to be competitive on the much smaller ADIC corpus (minimally outperforming state of the art) but failed to achieve competitive performance on GDIC (where the winning system HeLI was proposed by Jauhiainen *et al.* (2018a)).

We can speculate why this is the case. The results of the error analysis indicate a deterioration in performance of the proposed system on shorter documents. On the other hand, results of the VarDial 2018 shared tasks suggest that the performance of the HeLI system deteriorates less on shorter texts in comparison to other systems participating in shared tasks, since it ranked first on GDIC, where the documents are on average nine words long, and in the Vardial 2018 ILI shared task, where the task was to classify sentences[o], but only ranked fifth in the VarDial 2018 Discriminating between Dutch and Flemish in Subtitles task where the average document was 34.64 words long. Another hypothesis is that the proposed system is more reliant on named entities than the HeLI system, and therefore performs worse on GDIC, since this is the only corpus that does not contain news excerpts or news channel transcripts but transcripts of interviews with the dialect speakers and supposedly contains less named entities. We plan to test these hypotheses in the future work. We might also be able to boost the performance of our system on the GDIC data set by adjusting hyperparameters in order to make the network better suited for the classification of much shorter documents in the GDIC corpus, since currently a lot of data (e.g., *n*-grams that appear in less than five documents, character sequences filtered out by an aggressive max pooling ...) is discarded.

Small performance gains over the current state of the art also raise a question, how much better can automatic discrimination between similar languages actually get? The only study about the theoretical limit of the classification performance on the DSLCC that we are aware off was conducted by Goutte *et al.* (2016) on the DSLCC v2.0 used in the Vardial 2015 DSL shared task, which partially overlaps with the DSLCC v4.0 (Slavic, Malay and Indonesian, and Portuguese parts of the corpus are the same). First, they measured the upper bound on accuracy by taking all the predictions generated by all the systems which participated in the shared task and combining them using ensemble fusion methods such as plurality voting and Oracle. In the plurality voting, the label with most votes (i.e., the label predicted by most systems) is selected as correct and the conducted experiments showed that small improvements (of about 0.5 percentage point) over the best single system can be achieved. The Oracle method for determining the upper-bound performance on the other hand assigns the correct class label for an instance if at least one system classified the instance correctly. This gave them a very optimistic potential accuracy upper boundary of 99.83%.

In order to determine if the instances misclassified by the Oracle method can be correctly classified by humans, they conducted additional evaluation experiments. As it turns out, the difficulty of classification varies across different language groups. Discriminating between the three Slavic languages (Bosnian, Croatian, and Serbian) proved to be the most difficult. For 5 out of 12 instances misclassified by the Oracle method, none of the 6 annotators was able to correctly classify them. On these 12 examples the mean annotator accuracy was 16.66%, which is in fact 16.67% below the random baseline of 33.33%. On the other hand, discriminating between Brazilian and European Portuguese proved more feasible and the mean annotator accuracy on the misclassified instances was 67.50%, 17.50% above the 50% baseline.

This suggests that, at least for some varieties, the upper bound of automatic variety classification has not yet been reached, since our method achieved only 94.80% accuracy on the Portuguese language group. The conducted error analysis (see Section 5.3) on Slavic language varieties also showed that 10% of misclassified documents contained only clarifying named entities; therefore, any human annotator with some basic knowledge about Serbia, Bosnia, and Croatia would be able to classify them correctly without too much difficulty. This would suggest that further improvements on automatic language variety classification are possible, perhaps by employing

[o]We were unable to obtain the average document length for this data set since the number of tokens in the data set is not published in the Vardial 2018 report (Zampieri *et al.* 2018).

transfer learning techniques (Devlin *et al.* 2018) that would provide the classifier with the needed background information. We plan to test the transfer learning approach in the future.

CNNs have been so far the most successful neural architecture for language variety classification but the conducted ablation study shows that the produced features do have some deficiencies that make them less successful than weighted BON features. As shown, the proposed approach of feeding an additional weighted BON matrix into the network does partially compensate for these deficiencies on the language variety classification tasks but further work of exploring the synergy effects of combining automatically generated neural features and weighted features on a number of different NLP tasks and neural architectures is still needed. Feeding the sparse weighted BON matrix into the network does, however, have a drawback of drastically increasing the number of network parameters, which tends to lead to overfitting and increased computational costs. We managed to minimize these negative side effects mostly by an extensive use of dropout and by removing *n*-grams with low document frequencies from the input matrix, but perhaps a somewhat more efficient solution would be to avoid feeding the BON matrix to the neural classifier altogether. Therefore in future work, we plan to propose methods by which we would inject global document/corpus-level information into CNN-based features directly, in order to fix their current deficiencies. In that way combining them with the features that are the result of the more traditional feature engineering would no longer be required. Another option we also plan to explore is building heterogeneous ensembles of traditional SVM BOW-based models and CNNs and see if the performance gains are comparable to the proposed system.

Another line of future research will deal with building better and more useful tools for users with lower level of technical skills. Currently, the ClowdFlows platform does not support training of new neural classification models due to high level of resource consumption of these operations which would negatively affect the scalability of the platform, and since it does not yet support graphics processing unit (GPU) acceleration, which would allow for training of the models in a more reasonable time. The newer version of the ClowdFlows platform, on which the work has already begun, will address all these deficiencies and will allow for training of neural classification models on new varieties and therefore increase the overall usefulness of the system.

Acknowledgements. This paper is supported by European Union's Horizon 2020 research and innovation program under grant agreement No 825153—project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media) and grant agreement No 76966—project SAAM (Supporting Active Ageing through Multimodal coaching). The authors also acknowledge the financial support from the Slovenian Research Agency for research core funding for the program Knowledge Technologies (No P2-0103) and for the project TermFrame—Terminology and Knowledge Frames across Languages (No J6-9372). The results of this paper reflect only the author's view and the Commission is not responsible for any use that may be made of the information it contains. The Titan Xp used for this research was donated by the NVIDIA Corporation.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J. and Kudlur, M. (2016). Tensorflow: A system for large-scale machine learning. In *OSDI*, vol. 16, pp. 265–283.

Ali, A., Dehak, N., Cardinal, P., Khurana, S., Yella, S.H., Glass, J. and Renals, S. (2015). Automatic dialect detection in arabic broadcast speech. In *Proceedings of Interspeech*, pp. 2934–2938. San Francisco, USA: ISCA.

Ali, M. (2018a). Character level convolutional neural network for German dialect identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 172–177. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Ali, M. (2018b). Character level convolutional neural network for Arabic dialect identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 122–127. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Ali, M. (2018c). Character level convolutional neural network for Indo-Aryan language identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 283–287. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Alvarez-Carmona, M.A., López-Monroy, A.P., Montes-y-Gómez, M., Villasenor-Pineda, L. and Escalante, H.J. (2015). INAOE's participation at PAN'15: Author profiling task. In *Working Notes Papers of the CLEF*. Toulouse, France: CEUR Workshop Proceedings.

Belinkov, Y. and Glass, J. (2016). A character-level convolutional neural network for distinguishing similar languages and dialects. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 145–152. Osaka, Japan: The COLING 2016 Organizing Committee.

Basile, A., Dwyer, G., Medvedeva, M., Rawee, J., Haagsma, H. and Nissim, M. (2017). N-gram: New Groningen author-profiling model. In *CLEF 2017 Evaluation Labs and Workshop - Working Notes Papers*. Dublin, Ireland: CEUR Workshop Proceedings.

Bergstra, J., Bastien, F., Breuleux, O., Lamblin, P., Pascanu, R., Delalleau, O. and Bengio, Y. (2011). Theano: Deep learning on GPUs with python. In *NIPS 2011, BigLearning Workshop, Granada, Spain*, vol. 3, pp. 1–48.

Bestgen, Y. (2017). Improving the character n-gram model for the DSL task with BM25 weighting and less frequently used feature sets. In *proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 115–123. Valencia, Spain: Association for Computational Linguistics.

Bjerva, J. (2016). Byte-based language identification with deep convolutional networks. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 119–125. Osaka, Japan: The COLING 2016 Organizing Committee.

Chollet, F. (2015). Keras: Deep learning library for theano and tensorflow. https://keras.io.

Cianflone, A. and Kosseim, L. (2017). N-gram and neural language models for discriminating similar languages. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 243–250. Osaka, Japan: The COLING 2016 Organizing Committee.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K. and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research* **12**, 2493–2537.

Çöltekin, Ç. and Rama, T. (2016). Discriminating similar languages with linear SVMs and neural networks. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 15–24. Osaka, Japan: The COLING 2016 Organizing Committee.

Çöltekin, Ç., Rama, T. and Blaschke, V. (2018). Tübingen-Oslo team at the VarDial 2018 evaluation campaign: An analysis of n-gram features in language variety identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 55–65. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Criscuolo, M. and Aluisio, S.M. (2017). Discriminating between similar languages with word-level convolutional neural networks. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 124–130. Valencia, Spain: Association for Computational Linguistics.

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

Goutte, C., Léger, S. and Carpuat, M. (2014). The NRC system for discriminating similar languages. In *Proceedings of the First Workshop on Applying NLP Tools to Similar Languages, Varieties and Dialects*, pp. 139–145. Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Goutte, C., Léger, S., Malmasi, S. and Zampieri, M. (2016). Discriminating similar languages: Evaluations and explorations. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pp. 1800–1807. Portorož, Slovenia: European Language Resources Association.

Jauhiainen, T., Jauhiainen, H. and Lindén, K. (2018a). HeLI-based experiments in Swiss German dialect identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 254–262. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Jauhiainen, T., Jauhiainen, H. and Lindén, K. (2018b). Iterative language model adaptation for Indo-Aryan language identification. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 66–75. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T. (2016). Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pp. 427–431. Valencia, Spain: Association for Computational Linguistics.

Kingma, D.P. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*. San Diego, California, USA: DBLP.

Kranjc, J., Podpečan, V. and Lavrač, N. (2012). ClowdFlows: A cloud based scientific workflow platform. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 816–819. Bristol, UK: Springer.

López-Monroy, A.P., Montes-y-Gómez, M., Escalante, H.J. and Pineda, L.V. (2014). Using intra-profile information for author profiling. In *CLEF (Working Notes)*, pp. 1116–1120. Sheffield, UK: CEUR Workshop Proceedings.

Malmasi, S. and Dras, M. (2015). Language identification using classifier ensembles. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pp. 35–43. Hissar, Bulgaria: Association for Computational Linguistics.

Malmasi, S. and Zampieri, M. (2016a). Arabic dialect identification in speech transcripts. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 106–113. Osaka, Japan: The COLING 2016 Organizing Committee.

Malmasi, S., Zampieri, M., Ljubešić, N., Nakov, P., Ali, A. and Tiedemann, J. (2016b). Discriminating between similar languages and arabic dialect identification: A report on the third DSL shared task. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pp. 1–14. Osaka, Japan: The COLING 2016 Organizing Committee.

Martinc, M. and Pollak, S. (2018). Reusable workflows for gender prediction. In *Language Resources and Evaluation Conference (LREC 2018) Proceedings*, pp. 515–520. Miyazaki, Japan: European Language Resources Association.

McKinney, W. (2011). Pandas: A foundational Python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, pp. 1–9.

Miura, Y., Taniguchi, T., Taniguchi, M. and Ohkuma, T. (2017). Author profiling with word + character neural attention network. In *CLEF (Working Notes)*. Dublin, Ireland: CEUR Workshop Proceedings.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O. and Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. *Journal of machine learning research* **12**, 2825–2830.

Rangel, F., Rosso, P., Koppel, M., Stamatatos, E. and Inches, G. (2013). Overview of the author profiling task at PAN 2013. In *CLEF Conference on Multilingual and Multimodal Information Access Evaluation*, pp. 352–365. Valencia, Spain: Springer.

Rangel Pardo, F.M., Celli, F., Rosso, P., Potthast, M., Stein, B. and Daelemans, W. (2015). Overview of the 3rd author profiling task at PAN 2015. In *CLEF 2015 Evaluation Labs and Workshop Working Notes Papers*, pp. 1–8. Toulouse, France: CEUR Workshop Proceedings.

Rangel, F., Rosso, P., Verhoeven, B., Daelemans, W., Potthast, M. and Stein, B. (2016). Overview of the 4th author profiling task at PAN 2016: Cross-genre evaluations. In Balog, K. et al. (ed.) *Working Notes Papers of the CLEF 2016 Evaluation Labs*. CEUR Workshop Proceedings, pp. 750–784. Évora, Portugal: CEUR Workshop Proceedings.

Rangel, F., Rosso, P., Potthast, M. and Stein, B. (2017). Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter. In *Working Notes Papers of the CLEF*. Dublin, Ireland: CEUR Workshop Proceedings.

Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval* **3**(4), 333–389.

Samardzic, T., Scherrer, Y. and Glaser, E. (2016). Archimob-a corpus of spoken Swiss German. In *Proceedings of LREC 2016*, pp. 4061–4066. Portorož, Slovenia: European Language Resources Association.

Stamatatos, E., Daelemans, W., Verhoeven, B., Potthast, M., Stein, B., Juola, P. and Barrón-Cedeño, A. (2014). Overview of the author identification task at PAN 2014. In *CLEF 2014 Evaluation Labs and Workshop Working Notes Papers, Sheffield, UK, 2014*, pp. 1–21. Sheffield, UK: CEUR Workshop Proceedings.

Tan, L., Zampieri, M., Ljubešic, N. and Tiedemann, J. (2014). Merging comparable data sources for the discrimination of similar languages: The DSL corpus collection. In *Proceedings of the 7th Workshop on Building and Using Comparable Corpora (BUCC)*, pp. 11–15. Reykjavik, Iceland: European Language Resources Association.

Vollenbroek, M.B., Carlotto, T., Kreutz, T., Medvedeva, M., Pool, C., Bjerva, J. and Nissim, M. (2016). Gronup: Groningen user profiling. In *Notebook for PAN at CLEF*, pp. 846–857. Évora, Portugal: CEUR Workshop Proceedings.

Zampieri, M., Tan, L., Ljubešić, N. and Tiedemann, J. (2014). A report on the DSL shared task 2014. In *Proceedings of the first workshop on applying NLP tools to similar languages, varieties and dialects*, pp. 58–67. Dublin, Ireland: Association for Computational Linguistics and Dublin City University.

Zampieri, M., Tan, L., Ljubešić, N., Tiedemann, J. and Nakov, P. (2015). Overview of the DSL shared task 2015. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pp. 1–9. Hissar, Bulgaria: Association for Computational Linguistics.

Zampieri, M., Malmasi, S., Ljubešić, N., Nakov, P., Ali, A., Tiedemann, J. and Aepli, N. (2017). Findings of the VarDial evaluation campaign 2017. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pp. 1–15. Valencia, Spain: Association for Computational Linguistics.

Zampieri, M., Malmasi, S., Nakov, P., Ali, A., Shon, S., Glass, J. and Van der Lee, C. (2018). Language identification and morphosyntactic tagging: The second VarDial evaluation campaign. In *Proceedings of the Fifth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial 2018)*, pp. 1–17. Santa Fe, New Mexico, USA: Association for Computational Linguistics.

Cite this article: Martinc M and Pollak S. Combining *n*-grams and deep convolutional features for language variety classification. *Natural Language Engineering* **25**, 607–632. https://doi.org/10.1017/S1351324919000299

Appendix C: Multilingual Dynamic Topic Model

Multilingual Dynamic Topic Model

Elaine Zosa and **Mark Granroth-Wilding**
Department of Computer Science
University of Helsinki
Helsinki, Finland
firstname.lastname@helsinki.fi

1 Abstract

Dynamic topic models (DTMs) capture the evolution of topics and trends in time series data. Current DTMs are applicable only to monolingual datasets. In this paper we present the multilingual dynamic topic model (ML-DTM), a novel topic model that combines DTM with an existing multilingual topic modeling method to capture cross-lingual topics that evolve across time. We present results of this model on a parallel German-English corpus of news articles and a comparable corpus of Finnish and Swedish news articles. We demonstrate the capability of ML-DTM to track significant events related to a topic and show that it finds distinct topics and performs as well as existing multilingual topic models in aligning cross-lingual topics.

2 Introduction

Dynamic topic models (DTMs, Blei and Lafferty, 2006) capture themes or topics discussed in a set of time-stamped documents and how the words related to these topics change in prominence over time. Other topic models have been proposed that aim to model time series data (Wang and McCallum, 2006; Wei et al., 2007; Hall et al., 2008). These models can be used to explore historical document collections to study historical trends, language changes (Frermann and Lapata, 2016) and track the emergence and evolution of certain subjects (Hall et al., 2008; Yang et al., 2011).

With the internet becoming more multilingual it is increasingly important to build cross-lingual tools to bridge different linguistic groups online. Fortunately, large multilingual datasets such as Wikipedia, the Europarl parallel corpus (Koehn, 2005) and other datasets assembled from crawling the web (Van Gael and Zhu, 2007) are also becoming widely available to researchers. This has led to the development of several multilingual topic models to infer topics from multilingual datasets. Examples include the polylingual topic model (PLTM, Mimno et al., 2009), multilingual topic model for unaligned text (MuTo, Boyd-Graber and Blei, 2009), and JointLDA (Jagarlamudi and Daumé, 2010). What is currently lacking are topic models for multilingual time-stamped data that can model historical and linguistic changes in a specific context. Digitalization efforts in libraries and archives, such as the Europeana collections[1], have made available online historical document collections from different European countries. Collections such as these are valuable resources for comparing historical trends in different countries. However, scholars and other interested parties may not possess the linguistic skills necessary to explore such data and would benefit from tools to automatically discover connections across linguistic boundaries.

In this paper, we present the multilingual dynamic topic model (ML-DTM), a novel topic model that captures dynamic topics from broadly topically aligned multilingual datasets. We extend a DTM inference method by Bhadury et al. (2016) to train this model.

In the following sections, we give a broad review of related work, discuss existing *dynamic* and *multilingual* topic models in more detail, and then give a description of our proposed combined model. We then demonstrate usage of this model on a parallel dataset and a comparable dataset of news articles and present our results. We show that this novel topic model learns aligned bilingual topics as demonstrated by the cosine similarities of learned vector representations of named entities. Table 1 summarizes the notations used in this paper. Code is available at: https://github.com/ezosa/multilingual_dtm.

[1] https://www.europeana.eu

Symbol	Description
α	parameter for θ
β	hyperparameter for ϕ
ψ	hyperparameter for θ
θ	distribution of topics over a document
ϕ	distribution of words over a topic
D	set of documents
W_d	words in document d
N_d	number of words in document d, or $\|W_d\|$
Z_d	topic assignments of words in document d
K	number of topics
T	number of time slices
L	number of languages in the dataset
V	words in a vocabulary for language

Table 1: Summary of notations.

3 Related Work

Topic models capture themes inherent in document collections through the co-occurence patterns of the words in documents. Latent Dirichlet Allocation (LDA, Blei et al., 2003) is a popular method for inferring these themes or topics. It is generative document model where a document is described by a mixture of different topics and each topic is a probability distribution over the words in the vocabulary. In a document collection we can only observe the *words* in a document. Therefore, training a model involves inferring these latent variables through approximate inference methods.

In the case of documents with timestamps covering some time interval, such as news articles, we might want to capture *dynamic* co-occurence patterns that evolve through time. Dynamic Topic Model (DTM, Blei and Lafferty, 2006) divides time into discrete slices and chains parameters from each slice in order to infer topics that are aligned across time. DTM gives us a set of topic-term distributions that evolve from one time slice to the next. There are also other topic models for time-series data such as the Continuous Dynamic Topic Model (cDTM, Wang et al., 2008), a version of DTM that does not explicitly discretize time intervals. Dynamic Mixture Model (DMM, Wei et al., 2007) captures the evolution of documents across time and Topics over Time (TOT, Wang and McCallum, 2006) is a method that models the prominence of topics over time.

A limitation of LDA, as well as these dynamic models, is that it is not applicable to multilingual data. LDA captures co-occurences of words in documents and words from different languages would rarely, if ever, occur in the same document regardless of their semantics, as demonstrated by experiments on the Europarl corpus (Jagarlamudi and Daumé, 2010; Boyd-Graber and Blei, 2009). Multilingual topic models are developed to capture cross-lingual topics from multilingual datasets.

Polylingual Topic Model (PLTM, Mimno et al., 2009) is a multilingual topic model that extends LDA for an aligned multilingual corpus. Instead of running topic inference on individual documents as in LDA, PLTM infers topics for *tuples* of documents, where each document in the tuple is in a different language. PLTM assumes that the documents of a tuple discuss the same subject broadly and therefore share the same document-topic distribution.

Other topic models for multilingual data include Multilingual Topic Model for Unaligned Text (MuTo, Boyd-Graber and Blei, 2009) and JointLDA (Jagarlamudi and Daumé, 2010). MuTo attempts to match words between languages in the corpus and samples topic assignments for these matchings. JointLDA is a multilingual model that does not require an aligned corpus but requires a bilingual dictionary and uses concepts, instead of words, to infer topics where concepts can be entries in the bilingual dictionary.

In this work we will focus on DTM and PLTM because we want to capture topic evolution in multilingual settings without using additional lexical resources such as dictionaries.

3.1 Dynamic Topic Model

LDA uses Dirichlet and multinomial distributions for inferring both topic-term distributions ϕ and document-topic distributions θ. The conjugacy of these distributions allow ϕ and θ to be integrated out leaving us only with the posterior distribution for topic-term assignments Z, which we can sample through Gibbs sampling (Griffiths and Steyvers, 2004). Inference in DTM, however, is

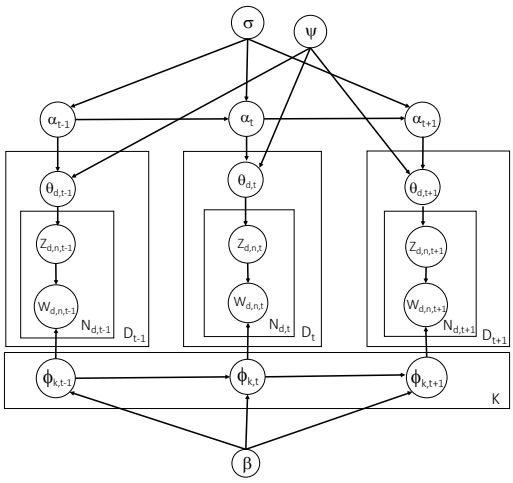


Figure 1: DTM for three time slices as shown in Bhadury et al. (2016).

more complicated due to the non-conjugacy of the distributions used in the model. Blei and Lafferty (2006) use variational Kalman filtering for topic inference, which does not scale well for a large number of topics and documents and large numbers of time slices (Bhadury et al., 2016; Wang et al., 2008). Bhadury et al. (2016) developed a method for inferring the posterior distributions of DTM with Gibbs sampling. In their method, the parameters α, θ, ϕ and Z are re-sampled during every iteration of the sampler.

The document-topic proportions θ, sampled for each document in each time slice, and the topic-term distributions ϕ, sampled for each topic in each time slice, are updated using Stochastic Gradient Langevin Dynamics (SGLD, Welling and Teh, 2011) which is based on Stochastic Gradient Descent (SGD). Figure 1 shows the plate diagram for DTM from Bhadury et al. (2016).

3.2 Polylingual Topic Model

The polylingual topic model (PLTM, Mimno et al., 2009) is an extension of LDA that infers topics from an aligned multilingual corpus composed of document tuples. Tuples are composed of documents in different languages that are thematically aligned, meaning that they discuss the subject in broadly similar ways. For instance, a news article in German and another article in English that report on the same event can compose a tuple.

Inference on PLTM can be done via Gibbs sampling where the topic assignment of each term $z^l_{d,n}$ is resampled during every iteration. Following Vulić et al. (2015), we provide the update formulae for the bilingual case for brevity. The update formulae for documents in languages x and y are:

$$P(z^x_{d,n} = k|z^x, z^y, w^x, w^y, \alpha, \beta) \propto \frac{m^x_{d,k} - 1 + m^y_{d,k} + \alpha}{\sum_{i=1}^{K} m^x_{d,i} - 1 + \sum_{i=1}^{K} m^y_{d,i} + K\alpha} \cdot \frac{v^x_{k,w_{d,n}} - 1 + \beta}{\sum_{i=1}^{|V^x|} v^x_{k,w_{d,i}} - 1 + |V^x|\beta} \quad (1)$$

$$P(z^y_{d,n} = k|z^y, z^x, w^y, w^x, \alpha, \beta) \propto \frac{m^y_{d,k} - 1 + m^x_{d,k} + \alpha}{\sum_{i=1}^{K} m^y_{d,i} - 1 + \sum_{i=1}^{K} m^x_{d,i} + K\alpha} \cdot \frac{v^y_{k,w_{d,n}} - 1 + \beta}{\sum_{i=1}^{|V^y|} v^y_{k,w_{d,i}} - 1 + |V^y|\beta} \quad (2)$$

where $m^x_{d,k}$ is the number of times topic k has been assigned to a word in document d written in language x and $v^x_{k,w_{d,n}}$ is the number of times word $w_{d,n}$, that is, the word at position n in document d, has been assigned to topic k. $|V^x|$ is the vocabulary size of language x. The first part of these formulae links the two languages together and is language-independent while the second part is language-specific.

Figure 2 shows the graphical representation of PLTM for l languages.

4 Multilingual Dynamic Topic Model

Here we combine the above *dynamic* and *polylingual* models to produce a *Multilingual Dynamic Topic Model* (ML-DTM). Figure 3 shows the diagram of ML-DTM for two languages and three time slices. Although we show only the bilingual case here for brevity, the model is applicable for any number of languages.

The inference method of Bhadury et al. (2016) was originally motivated by the need to speed up DTM inference for very large datsets. We apply it here to the combined ML-DTM model. We propose the following posterior conditional distribution for $\theta_{x,t}$ where x is a tuple index in the dataset:

$$p(\theta_{x,t}|\alpha_t, Z_{x,t}) \propto \mathcal{N}(\theta_{x,t}|\alpha_t, \psi^2 I) \times \prod_{l=1}^{L} \prod_{n=1}^{N_{d_l,t}} Mult(Z_{d_l,n,t}|\pi(\theta_{x,t})) \quad (3)$$

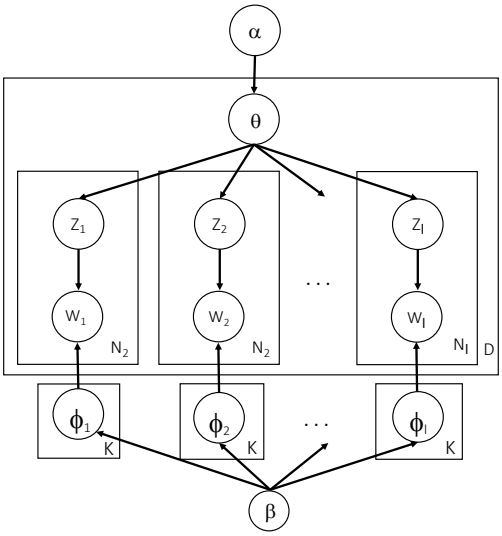


Figure 2: Polylingual topic model for l languages of Mimno et al. (2009).

Following Bhadury et al. (2016), the update equation to evaluate the gradient of $\theta^k_{x,t}$ becomes:

$$\nabla_{\theta^k_{x,t}} \log p(\theta_{x,t}|\alpha_t, Z_{x,t}) = \frac{-1}{\psi^2}(\theta^k_{x,t} - \alpha^k_t) + \sum_{l=1}^{L} C^k_{d_l,t} - \left(N_{d_l,t} \times \frac{\exp(\theta^k_{x,t})}{\sum_j \exp(\theta^j_{x,t})} \right) \quad (4)$$

where $Z_{x,t}$ are the topic assignments for the words in the documents in tuple x at time slice t; $C^k_{d_l,t}$ is the number of times topic k has been assigned to a word in document d_l at time t; and $N_{d_l,t}$ is the length of document d_l at time t.

Instead of evaluating $\theta_{d,t}$ for a single document as in monolingual DTM, we compute $\theta_{x,t}$ for a document *tuple*. The second term in (4) links the languages together by summing up the counts of each document in the tuple.

The equation for evaluating the gradient of the topic-term distributions $\phi_{k,t}$ is the same as in the original paper except that we compute separate distributions for each language since every language has a different vocabulary. This means that for each time slice, instead of updating K different ϕs (one for each topic), we will need to update $K \cdot L$ ϕs. Table 2 shows the dimensions of the parameters to be estimated.

Finally, the topic assignment $Z_{d_l,n,t}$ is sampled

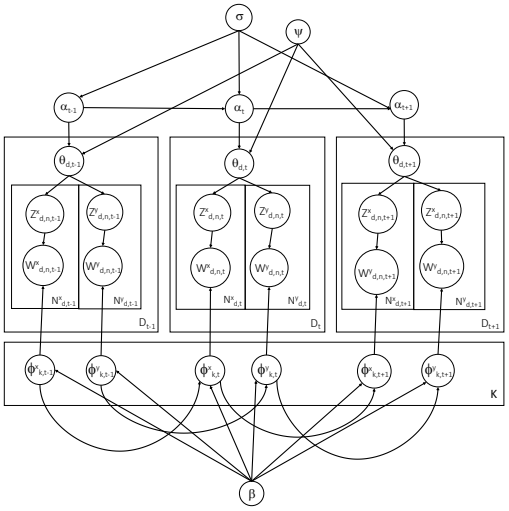


Figure 3: ML-DTM for two languages and three time slices.

Parameter	Dimension
α	$K \times T$
θ	$D^t \times K \times T$
ϕ	$\lvert V^l \rvert \times L \times K \times T$

Table 2: Dimensions of the sampled parameters in the multilingual dynamic topic model (ML-DTM). D^t is the number of document tuples in a dataset.

as in the original paper:

$$P(Z_{d_l,n,t} = k|\theta_{x,t}, \phi^{w_l}_{k,t}) \propto exp(\theta^k_{x,t}) exp(\phi^{w_l}_{k,t}) \quad (5)$$

where w_l is a word from the vocabulary of language l.

5 Evaluation

5.1 Datasets

We ran experiments on ML-DTM with two kinds of data: a parallel dataset and a thematically-comparable one.

The DE-NEWS parallel dataset consists of German news articles from August 1996 to January 2000 with English translations done by human volunteers[2]. This dataset covers 42 months with an average of 200 articles per month. Since this is a parallel corpus there is no need to align the articles.

[2] http://homepages.inf.ed.ac.uk/pkoehn/publications/de-news/

For the comparable dataset, we use the YLE news dataset which consists of Finnish and Swedish articles from the Finnish broadcaster YLE, covering news in Finland from January 2012 to December 2018[3]. The Finnish and Swedish articles are written separately and are not direct translations of each other. We use existing methods for aligning comparable news articles (Utiyama and Isahara, 2003; Vu et al., 2009). Specifically, we create an aligned corpus by pairing a Finnish article with a Swedish article published within a two-day window and sharing three or more named entities. We want to have a one-to-one alignment in our dataset such that no article is duplicated, so we pair a Finnish article with the first Swedish article encountered in the dataset that fits the above criteria and remove the paired articles from the unaligned dataset. The unaligned dataset has a total of 604,297 Finnish articles and 228,473 Swedish articles and the final aligned dataset consists of 123,818 articles covering 84 months. A script for aligning articles using the method described is provided in the Github project associated with this work.

We tokenized, lemmatized (using WordNetLemmatizer for German and English and LAS (Mäkelä, 2016) for Finnish and Swedish) and removed stopwords for these two datasets and then used the 5,000 most frequent words of each language as the vocabulary for that language.

5.2 Cross-Lingual Alignment

We compare the cross-lingual alignment of topics of ML-DTM and PLTM by evaluating the similarity of the learned vector representations of named entities (NEs) that appear in both languages of the same dataset. This method is suggested by Vulić et al. (2015) on the basis that NEs tend to be spelled in the same way in different languages and can be expected to have a similar association with topics across languages. The K-dimensional vector of a NE w for language s is thus:

$$vec(w_s) = [P(z_1|w_s), P(z_2|w_s), ..., P(z_K|w_s)] \quad (6)$$

Under an assumption of a uniform prior over topics, this vector can be computed as:

$$P(z_k|w_s) \propto \frac{P(w_s|z_k)}{P(w_s)} = \frac{\phi_{l,z_k,w_s}}{Norm_{\phi_{s,.,w_s}}} \quad (7)$$

$$Norm_{\phi_{s,.,w_s}} = \sum_{k=1}^{K} \phi_{s,z_k,w_s} \quad (8)$$

$$vec(w_s) = \frac{[\phi_{l,z_1,w_s}, \phi_{l,z_2,w_s}, ..., \phi_{l,z_K,w_s}]}{Norm_{\phi_{s,.,w_s}}} \quad (9)$$

We then take the cosine similarities between the L different vector representations of the NE (for both datasets, $L = 2$).

We evaluate the cosine similarities of NEs that occur five or more times in each time slice. To make the comparison between PLTM and ML-DTM, we train one ML-DTM model on three time slices for 10 topics and three separate PLTM models for each time slice, also capturing 10 topics. We set $\alpha = 1.0$ and $\beta = 0.08$ for PLTM and $\alpha = 0.5$ and $\beta = 0.5$ for ML-DTM for both datasets, which achieved the best results of a small range of values tried. We did not, for now, perform more extensive optimisation of hyperparameters.

5.3 Topic Diversity

We also measure the *diversity* of the topics ML-DTM finds by computing the Jensen-Shannon (JS) divergence of every topic pair for each time slice for each language and averaging the divergences. Wang and McCallum (2006) used this method, though with KL divergence. It is desirable for the model to find topics that are as distinct as possible from each other.

We compare the diversity of the topics found by ML-DTM, trained as in the previous section, with the topics found by DTM. To make this comparison we train separate DTM models for each language in our two datasets, giving us four different models and compare the divergences of the topics found by these models with their ML-DTM counterparts. We use the Gensim implementation of DTM[4] where we set the chain variance to 0.1 and leave other parameters to be inferred during training. We train both ML-DTM and DTM on 10 time slices for 10 topics.

[3] https://www.kielipankki.fi/corpora/

[4] https://radimrehurek.com/gensim/models/ldaseqmodel.html

Time slice	# of NEs	PLTM	ML-DTM
Aug 1996	53	**0.880**	0.692
Sept 1996	65	0.876	**0.908**
Oct 1996	64	0.840	**0.885**

Table 3: Average cosine similarity of topic vectors for NEs over three time slices in DE-NEWS.

Time slice	# of NEs	PLTM	ML-DTM
Jan 2012	79	0.800	**0.896**
Feb 2012	71	**0.810**	0.796
Mar 2012	72	0.722	**0.745**

Table 4: Average cosine similarity of the vectors of NEs for three time slices in the YLE dataset.

6 Results and Discussion

Tables 3 and 4 show the average cosine similarity between NEs for each language in the DE-NEWS and YLE datasets, respectively. In the DE-NEWS data (Table 3), PLTM outperforms ML-DTM in the first time slice but ML-DTM performs better on the succeeding time slices. This is an encouraging result, considering that the parameters of ML-DTM at time slice t are estimated from adjacent time slices, adding a large degree of complexity to the model, whereas PLTM estimates parameters based on the current time slice only (PLTM has no concept of time).

For the YLE dataset (Table 4), ML-DTM shows an improvement in the first time and third slices and comparable performance in the second. The comparable nature of this dataset makes aligning NEs a more challenging task for both models. One way to improve performance on this task might be to use stricter criteria in aligning the dataset, such as pairing articles only if they were published on the same day or if they share more named entities.

We compare topic diversity of the topics found by DTM and ML-DTM. Tables 5 and 6 show the average JS divergence of every topic pair for five time slices in the DE-NEWS and YLE datasets, respectively. ML-DTM consistently learns more diverse topics than DTM for both datasets.

In Figure 4, we show the evolution of one topic found by ML-DTM trained on DE-NEWS. We show the top words of a topic about labor unions for the first eight months of the dataset. The English and German words are not exact translations of each other but we see similar or related words

Time slice	DTM English	ML-DTM English
Aug 1996	0.372	**0.655**
Sep 1996	0.368	**0.660**
Oct 1996	0.366	**0.657**
Nov 1996	0.365	**0.664**
Dec 1996	0.363	**0.650**
	DTM German	ML-DTM German
Aug 1996	0.315	**0.661**
Sep 1996	0.312	**0.670**
Oct 1996	0.310	**0.665**
Nov 1996	0.308	**0.638**
Dec 1996	0.306	**0.666**

Table 5: Topic diversity comparison between DTM and ML-DTM: average JS divergences of each topic pair for five months of the DE-NEWS dataset for English and German.

and NEs in each time slice. For instance, in August 1996 'employer' and 'arbeitgeber' both appear, as does 'einzelhandel' and 'retail'. In Sept 1996, 'kohl' is the top term for both languages (referring to former German chancellor Helmut Kohl). There are cases where German terms have no direct translation in English but an equivalent concept appears in the English topic. This is the case with 'lohnfortzahlung' (sick-leave pay) where the terms 'sick' and 'pay' appear on the English side; and 'steuerreform' (tax reform) where 'reform' appears on the English side as well.

A named entity, 'thyssen', appears in March 1997 in both languages but not in other months. This is because of an event that happened around mid-March where the German steel company Thyssen was being bought by competitor Krupp-Hoesch (also a top term in the German topic) prompting concerns about job losses[5].

Figure 5 shows the first six months of a topic about political news from the YLE dataset. The first two months has terms related to presidential elections. This refers to the Finnish presidential election in 2012, where rounds of voting took place in January and February 2012[6]. These time slices also mention the two candidates in the runoff election, Sauli Niinistö and

[5] https://www.nytimes.com/1997/03/19/business/krupp-hoesch-confirms-bid-of-8-billion-for-thyssen.html

[6] https://en.wikipedia.org/wiki/2012_Finnish_presidential_election

Time slice	DTM Finnish	ML-DTM Finnish
Jan 2012	0.332	**0.445**
Feb 2012	0.324	**0.465**
Mar 2012	0.322	**0.470**
Apr 2012	0.353	**0.498**
May 2012	0.357	**0.495**
	DTM Swedish	ML-DTM Swedish
Jan 2012	0.365	**0.480**
Feb 2012	0.360	**0.491**
Mar 2012	0.354	**0.497**
Apr 2012	0.388	**0.535**
May 2012	0.393	**0.537**

Table 6: Topic diversity comparison between DTM and ML-DTM: average JS divergences of each topic pair for five months of the YLE dataset for Finnish and Swedish.

Pekka Haavisto. Sauli Niinistö eventually won the election which explains why the next time slices ceases to mention Pekka Haavisto while 'niinistö' is still a prominent term. After March 2012, the topic stops talking about presidential elections and moves on to other political news. This gives us an insight into how the model can track significant events, such as high-profile elections, related to a topic. Another example is May 2012, where Greece ('kreikka' in Finnish, 'grekland' in Swedish) suddenly becomes a prominent term for both languages due to the Greek legislative elections which took place on 6 May 2012. The term 'syyria'/'syrien' appears in May and June, corresponding to the beginning of the Syrian Civil War.

Figure 6 shows the posterior probabilities of some terms related to the presidential elections ('niinistö'), Greece ('kreikka' or 'grekland') and Syria ('syyria' or 'syrien') in the political news topic for both languages. We see the rise and fall of the prominence of the terms according to their relevance in the news.

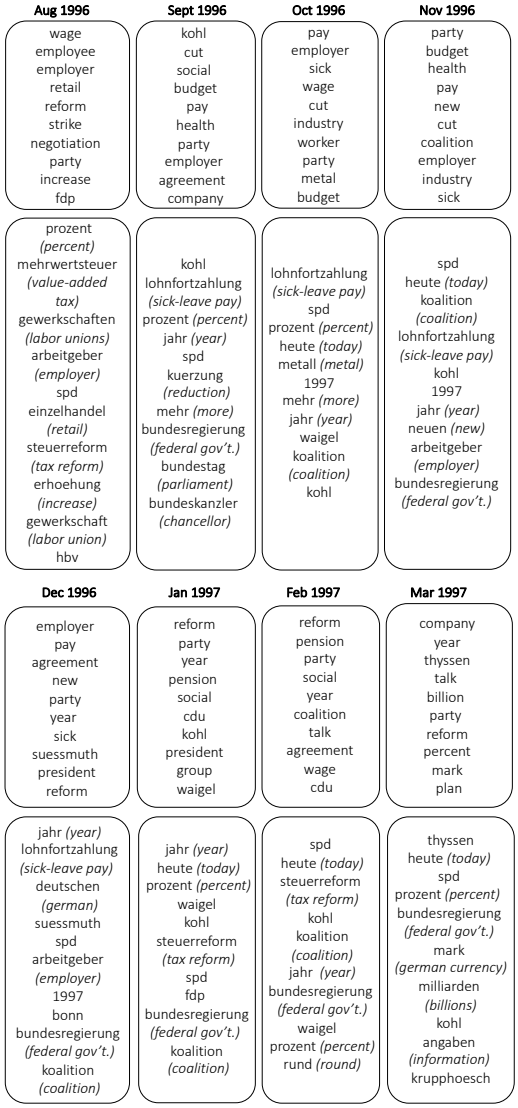


Figure 4: Top words of a topic concerning news about labor unions from the DE-NEWS dataset for English (top) and German (bottom) from Aug 1996 to March 1997. English translations of the German words excluding named entities are enclosed in parentheses.

7 Conclusions and Future Work

In this paper we present a novel topic model, the *multilingual dynamic topic model* (ML-DTM), that combines dynamic topic modeling (DTM) and polylingual topic modeling (PLTM) to infer dynamic topics from aligned multilingual data. ML-DTM uses an extension of the DTM inference method of Bhadury et al. (2016) to aligned multilingual data.

We ran experiments on ML-DTM with parallel and comparable datasets. We compare cross-lingual topic alignment of PLTM and ML-DTM by evaluating the cosine similarities of topic vectors corresponding to named entity terms across languages for corresponding time slices. ML-DTM achieves similar performance to PLTM on DE-NEWS and the comparable dataset (YLE). We also demonstrate the ability of ML-DTM to detect

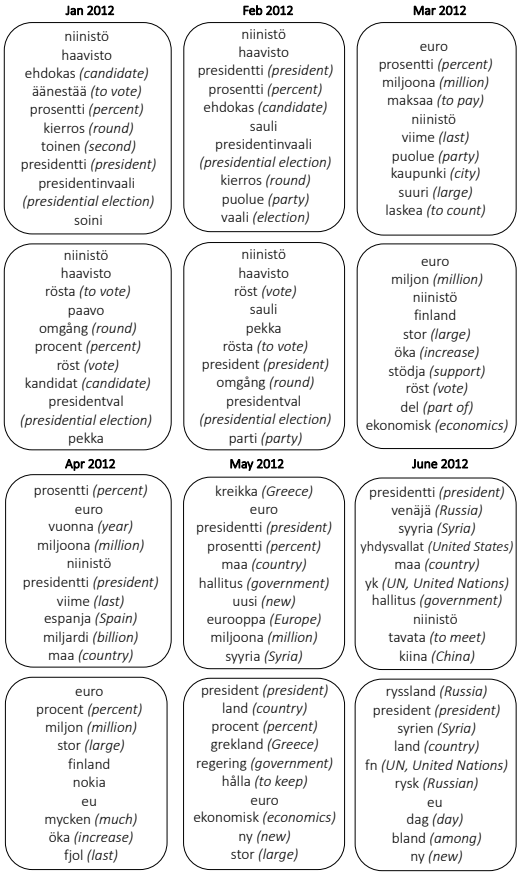


Figure 5: Top words of a topic on political news in Finland from the YLE dataset for Finnish (top) and Swedish (bottom) from Jan to June 2012. English translations of the words excluding named entities are enclosed in parentheses.

significant events regarding a topic through sudden changes in the prominent terms of the topic. This same method can also detect approximately when the event emerged and when it ended.

In a further experiment, we compared ML-DTM to the monolingual DTM, showing that ML-DTM achieves a consistently higher topic diversity within a single language.

We plan to run further experiments with ML-DTM using noisy datasets, such as historical news data where OCR errors might affect upstream tasks such as tokenization and lemmatization. We also plan to use named-entity recognition to improve our model such that named entities are treated as distinct items in the model's vocabulary, allowing us to track mentions of an entity across time slices and languages.

Historical news data covering a longer time

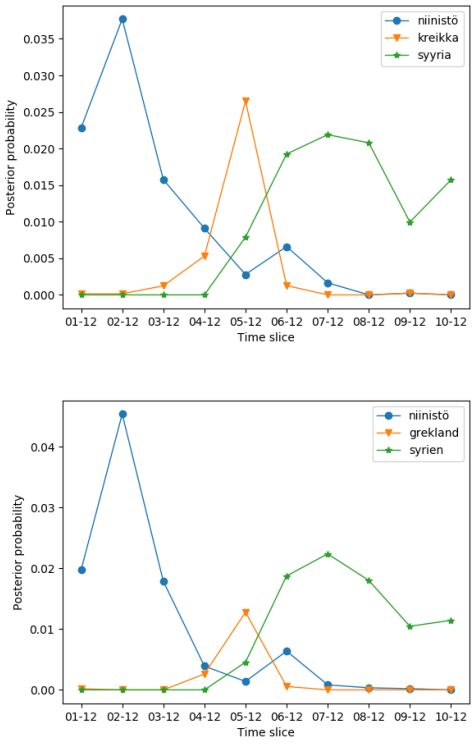


Figure 6: Posterior probabilities of salient terms in Finnish (top) and Swedish (bottom) related to events in the political news topic captured by ML-DTM from the YLE dataset.

span (several decades or more) would also enable us to study the changes in the use of words in a language and compare these changes with other languages. Historical news data from different regions would enable us to compare the way certain historical events were discussed in these places.

Acknowledgements

This work has been supported by the European Union's Horizon 2020 research and innovation programme under grants 770299 (NewsEye) and 825153 (EMBEDDIA).

References

Arnab Bhadury, Jianfei Chen, Jun Zhu, and Shixia Liu. 2016. Scaling up dynamic topic models. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, pages 381–390.

David M Blei and John D Lafferty. 2006. Dynamic topic models. In *Proceedings of the 23rd interna-*

tional conference on Machine learning. ACM, pages 113–120.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan):993–1022.

Jordan Boyd-Graber and David M Blei. 2009. Multilingual topic models for unaligned text. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, pages 75–82.

Lea Frermann and Mirella Lapata. 2016. A bayesian model of diachronic meaning change. *Transactions of the Association for Computational Linguistics* 4:31–45.

Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences* 101(suppl 1):5228–5235.

David Hall, Daniel Jurafsky, and Christopher D Manning. 2008. Studying the history of ideas using topic models. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, pages 363–371.

Jagadeesh Jagarlamudi and Hal Daumé. 2010. Extracting multilingual topics from unaligned comparable corpora. In *European Conference on Information Retrieval*. Springer, pages 444–456.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT summit*. volume 5, pages 79–86.

Eetu Mäkelä. 2016. Las: an integrated language analysis tool for multiple languages. *The Journal of Open Source Software* 1.

David Mimno, Hanna M Wallach, Jason Naradowsky, David A Smith, and Andrew McCallum. 2009. Polylingual topic models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*. Association for Computational Linguistics, pages 880–889.

Masao Utiyama and Hitoshi Isahara. 2003. Reliable measures for aligning japanese-english news articles and sentences. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, pages 72–79.

Jurgen Van Gael and Xiaojin Zhu. 2007. Correlation clustering for crosslingual link detection. In *IJCAI*. pages 1744–1749.

Thuy Vu, Ai Ti Aw, and Min Zhang. 2009. Feature-based method for document alignment in comparable news corpora. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 843–851.

Ivan Vulić, Wim De Smet, Jie Tang, and Marie-Francine Moens. 2015. Probabilistic topic modeling in multilingual settings: An overview of its methodology and applications. *Information Processing & Management* 51(1):111–147.

Chong Wang, David Blei, and David Heckerman. 2008. Continuous time dynamic topic models. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, Arlington, Virginia, United States, UAI'08, pages 579–586. http://dl.acm.org/citation.cfm?id=3023476.3023545.

Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pages 424–433.

Xing Wei, Jimeng Sun, and Xuerui Wang. 2007. Dynamic mixture models for multiple time-series. In *Ijcai*. volume 7, pages 2909–2914.

Max Welling and Yee W Teh. 2011. Bayesian learning via Stochastic Gradient Langevin Dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. pages 681–688.

Tze-I Yang, Andrew Torget, and Rada Mihalcea. 2011. Topic modeling on historical newspapers. In *Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities*. pages 96–104.

Appendix D: Towards Robust Text Classification with Semantics-Aware Recurrent Neural Architecture

machine learning & knowledge extraction

MDPI

Article

Towards Robust Text Classification with Semantics-Aware Recurrent Neural Architecture

Blaž Škrlj [1,2], Jan Kralj [1], Nada Lavrač [1,3] and Senja Pollak [1,4,*]

1 Jožef Stefan Institute, 1000 Ljubljana, Slovenia; blaz.skrlj@ijs.si (B.Š.); jan.kralj@ijs.si (J.K.); nada.lavrac@ijs.si (N.L.)
2 Jožef Stefan International Postgraduate School, 1000 Ljubljana, Slovenia
3 School of Engineering and Management, University of Nova Gorica, 5000 Nova Gorica, Slovenia
4 Usher Institute, Medical School, University of Edinburgh, Edinburgh EH16 4UX, UK
* Correspondence: senja.pollak@ijs.si

Received: 21 February 2019; Accepted: 1 April 2019; Published: 4 April 2019

check for updates

Abstract: Deep neural networks are becoming ubiquitous in text mining and natural language processing, but semantic resources, such as taxonomies and ontologies, are yet to be fully exploited in a deep learning setting. This paper presents an efficient semantic text mining approach, which converts semantic information related to a given set of documents into a set of novel features that are used for learning. The proposed Semantics-aware Recurrent deep Neural Architecture (SRNA) enables the system to learn simultaneously from the semantic vectors and from the raw text documents. We test the effectiveness of the approach on three text classification tasks: news topic categorization, sentiment analysis and gender profiling. The experiments show that the proposed approach outperforms the approach without semantic knowledge, with highest accuracy gain (up to 10%) achieved on short document fragments.

Keywords: recurrent neural networks; text mining; semantic data mining; taxonomies; document classification

1. Introduction

The task of classifying data instances has been addressed in data mining, machine learning, database, and information retrieval research [1]. In text mining, document classification refers to the task of classifying a given text document into one or more categories based on its content [2]. A text classifier is given a set of labeled documents as input, and is expected to learn to associate the patterns appearing in the documents to the document labels. Lately, deep learning approaches have become a standard in natural language-related learning tasks, showing high performance in different classification tasks involving various text types, including sentiment analysis of tweets [3] and news categorization [4].

Semantic data mining denotes a data mining approach where (domain) ontologies are used as background knowledge in the data mining process [5]. Semantic data mining approaches have been successfully applied in semantic subgroup discovery [6], data visualization [7], as well as text classification [8,9]. Provision of semantic information allows the learner to use features on a higher semantic level, allowing for data generalization. The semantic information is commonly represented as relational data in the form of networks or ontologies. Even though there are many sources of such knowledge, approaches capable of leveraging such information in a deep learning setting are still scarce.

This paper proposes a novel approach where semantic information in the form of taxonomies (i.e., ontologies with only hierarchical relations) is propositionalized and then used in a recurrent neural

network architecture. The proposed SRNA (Semantics-aware Recurrent Neural Architecture) approach has been tested on a document classification task, while special attention is paid to the robustness of the method on short document fragments. Classification of short or incomplete documents is useful in a large variety of tasks. For example, in author profiling, the task is to recognize author's characteristics, such as age or gender [10], based on a collection of author's text samples, where the effect of data size is known to be an important factor influencing classification performance [11]. A frequent text type for this task are tweets, where a collection of tweets from the same author is considered a single document, to which a label must be assigned. The fewer instances (tweets) we need, the more powerful and useful is the approach. In a similar way, this holds true for nearly any kind of text classification task. For example, for labeling a news article with a topic tag, using only snippets or titles and not the entire news, may be preferred due to limited text availability or required processing speed.

It has been demonstrated that deep neural networks need a large amount of information in order to learn complex representations from text documents, and that state-of-the-art models do not perform well when incomplete information is used as input [12]. This work addresses an open problem of increasing the robustness of deep neural network-based classifiers in such settings by exploring to what extent the documents can be truncated without affecting the learner's performance.

This work is structured as follows. Section 2 presents the background and related work. Section 3 introduces the proposed SRNA architecture, where semantic information in the form of taxonomies is propositionalized and used in a recurrent neural architecture. Sections 4 and 5 present the experimental setup and results of the evaluation on three publicly available data sets, with a special focus on how the constructed semantic vectors affect the classifier's performance. We conclude the paper in Section 6 with the plans for further work.

2. Background and Related Work

This section outlines the background and the related work in semantics-aware data mining and deep learning architectures.

2.1. Document Representation and Semantic Context

Document classification is highly dependent on *document representation*. In simple bag-of-words representations, the frequency (or a similar weight such as term frequency inverse document frequency) of each word or n-gram is considered as a separate feature. More advanced representations group words with similar meaning together. The approaches include Latent Semantic Analysis (LSA) [13], Latent Dirichlet Allocation (LDA) [14], and more recently word embeddings [15], which transform data instances (documents) into feature vectors in a lower-dimensional numeric vector space. One of the well known algorithms for word embedding is word2vec [15], which uses a two-layer shallow neural network architecture to capture the word context of the given text. As word2vec captures limited contextual information, recently introduced embedding approaches such as GloVe [16] and FastText [17] attempt to address these issues. Individual embeddings (feature vectors) are positioned closer if they are contextually more similar. Both embedding and LSA-based approaches have significantly improved in the recent years, both in terms of scalability, as well as in terms of their predictive power [18,19].

It has been previously demonstrated that context-aware algorithms significantly outperform the naive learning ones [20]. Neural networks can learn word representations by using their context, and are as such especially useful for text classification tasks. We refer to such semantic context as the *first-level context*.

Second-level context can be introduced by incorporating extensive amounts of *background knowledge* (e.g., in the form of ontologies or taxonomies) into a learning task, which can lead to improved performance of semantics-aware rule learning [6], subgroup discovery [21], and random forest learning [22]. In text mining, Elhadad et al. [23] report an ontology-based web document classifier,

while Kaur et al. [24] propose a clustering-based algorithm for document classification, which also benefits from the knowledge stored in the underlying ontologies.

Cagliero and Garza [20] report a custom classification algorithm, which can leverage taxonomies, and demonstrate—on a case study of geospatial data—that such information can be used to improve classification. Use of hypernym-based features for classification tasks has been considered previously. The Ripper rule learner was used with hypernym-based features [8], while the impact of WordNet-based features for text classification was also evaluated [9], demonstrating that hypernym-based features significantly impact the classifier performance.

Even though including background information in deep learning has yet to be fully exploited, there are already some *semantic deep learning* approaches available for text classification. Tang et al. [19] have demonstrated that word embedding approaches can take into account semantics-specific information to improve classification. Ristoski et al. [25] show that embedding-based approaches are useful for taxonomy induction and completion. Liu et al. [26] address incorporation of taxonomy-derived background knowledge as a constrained optimization problem, demonstrating that semantic information can be valuable for the tasks of entity recognition and sentence completion. Finally, Bian et al. [27] leverage morphological, syntactic, and semantic knowledge to achieve high-quality word embeddings and prove that knowledge-powered deep learning can enhance their effectiveness.

2.2. Deep Learning Architectures

This section introduces *deep learning architectures* for text classification.

A two-layer neural network has been introduced as part of the word2vec embedding approach [15]. Recently, deeper architectures have proven to work well in document classification tasks [28–30], where a neural network is given a set of vectors, whose elements are e.g., individual word indexes that are directly used to produce class predictions. These approaches include *convolutional neural networks*, which have been previously proven to work well for image classification [31,32]. A convolution is defined as:

$$s(t) = (x * w)(t) = \sum_{m=-\infty}^{\infty} x(m)w(t-m),$$

where x is the input function, m the input vector dimensionality and w is a kernel.

Kernels are smaller sub-matrices, which are applied in the process of convolution, and result in a modified origin matrix that can represent e.g., an image or a text sequence.

A convolutional neural network consists of at least three different types of computational layers: a convolution layer, a pooling layer, and a dense fully connected layer. The convolution layer returns convolutions computed on the given (single or multidimensional) inputs. Such a layer is normally followed by a pooling layer. Here, sets of neurons' outputs are merged into a single real number r. Common pooling layers include maximum and average pooling. Finally, the fully connected layer consists of a set of neurons, such that each neuron in the fully connected layer is connected with each neuron in the previous layer. In most contemporary convolutional architectures, fully connected layers (the first types of layers to be used in neural networks) are only used in the final stages due to their prohibitive computational cost. Single-dimensional convolutional networks are used extensively in natural language processing (NLP) tasks [28,33]. In a standard setting, vectors of word indexes are used as input for a deep learning-based text classifier. The first layer in such architectures is responsible for the construction of a lower-dimensional word index embedding, which is further used for learning. The objective of this layer is to project the high dimensional input into a lower dimensional vector space, more suitable for computationally expensive learning [34].

Recently, *recurrent neural networks* have gained significant momentum [35]. A recurrent neural network is a type of architecture with recurrent connections between individual neurons. Similarly to feedback loops in biology, such architectures to some extent enable memory storage. The most

commonly used recurrent architecture for sequence classification include the so-called Long-Short Term Memory (LSTM) cells [36] and Gated Recurrent Units (GRUs) [37].

A single LSTM cell consists of three main gates: the input, output and the forget gate (see Figure 1). Individual activations within a LSTM cell are defined as sigmoid functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}.$$

All three gates together form a feedback loop preserving gradients during the training. The main benefit for sequence learning is that LSTMs to some extent solve the vanishing gradient problem, i.e., long term signals remain in the memory, whereas a simple feedforward architecture is prone to vanishing gradients.

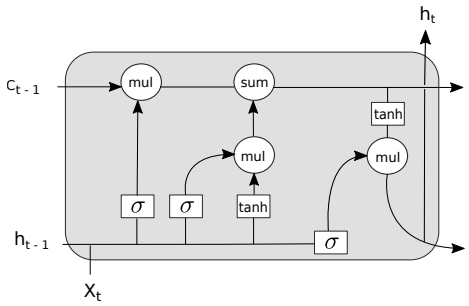


Figure 1. The LSTM cell. The forget gate is responsible for selective information filtering during the learning step [36,38]. Here, the C_{t-1} corresponds to the memory state at learning step $t-1$. We refer the interested reader to [38] for a more detailed overview of the LSTM cells shown here.

One issue common to all neural network models is that they often overfit the data. One of the most common solutions is the introduction of dropout layers [39] (at each training step, a percentage of neurons is omitted from being trained). We use them for regularization.

To achieve the state-of-the-art performance, sets of trained neural networks can be combined into neural ensembles. Some of the well known approaches which exploit this property include HDLTex [40] and RMDL [38]. Both approaches focus on learning of different aspects of the data set, yielding robust and powerful ensamble classification methods for e.g., text classification.

Large success of neural networks for classification is due to their capability of learning latent relationships in the data. In this work, we evaluate how additional information in the form of taxonomies affects the learning process. Even though feature engineering is becoming less relevant in the era of deep learning [41], we believe that integrating background knowledge can potentially improve classification models, especially when data is scarce, which is one of the currently unsolved problems related to deep architectures.

3. Proposed SRNA Approach

This section presents the proposed SRNA (Semantics-aware Recurrent Neural Architecture) approach, which leverages knowledge from taxomomies for construction of novel features for use in a custom deep neural network architecture. Figure 2 outlines the proposed two-step approach. In step 1 (described in Section 3.1), an input corpus $\mathcal{D}$ and a hypernym taxonomy are used to construct separate feature matrices D and S. In step 2 (described in Section 3.2), the two matrices are input into a hybrid neural network architecture to predict labels of new input documents.

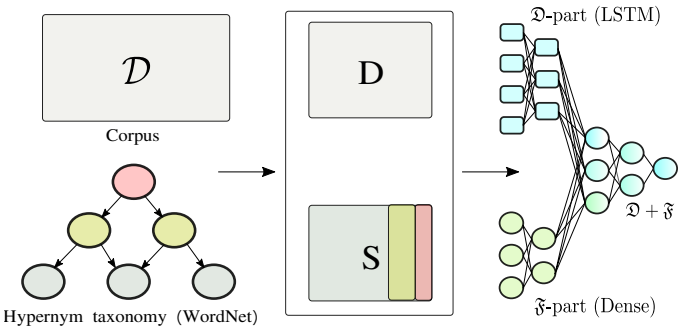


Figure 2. Visualization of the SRNA approach to semantic space propositionalization and learning. Left: A document corpus $\mathcal{D}$ and a hypernym taxonomy (WordNet). Middle: A matrix of word indexes D obtained from corpus $\mathcal{D}$, and a matrix of semantic features vectors S (with the same number of rows as D), with features obtained from different levels of the taxonomy. Right: A hybrid neural network architecture is learned from the word index vectors and the semantic feature vectors. Note that sequential word information is present only in the vectors constituting matrix D (word indices), hence part of the architecture exploits sequential information, whereas the constructed semantic features are input to the dense feedforward part of the architecture. Prior to the final layer, intermediary layers of both parts of the network are merged.

3.1. Propositionalization of the Semantic Space

The first step of the SRNA approach is hypernym identification and selection. We investigate how hypernyms can be used as additional background knowledge to possibly improve the classification. We rely on WordNet [42], a large and widely used lexical resource, in which words are annotated with word senses (i.e., word meanings) and connected by semantic relations, including synonymy (e.g., *car* $\leftrightarrow$ *auto*), hypernymy (e.g., *car* $\rightarrow$ *vehicle*) and hyponymy (e.g., *vehicle* $\rightarrow$ *car*). In this work, we explore only the space of hypernymy relations. The obtained hierarchical structure is thus a taxonomy. In order to leverage the extensive knowledge stored in word taxonomies, a propositionalization algorithm was developed, performing the fusion of the original set of documents $\mathcal{D}$, represented by word index matrix D of dimension $N \times \ell$ (is the user defined parameter for determining the dimension of their feature vectors, corresponding to the number of word indices used), with newly constructed semantic features. These features are the hypernyms, forming the columns of the semantic feature matrix S of dimension $N \times m$. The process of propositionalization merges (concatenates) the original matrix D and the sematic feature matrix S into novel matrix DS of dimension $N \times (\ell + m)$.

The semantic feature matrix S is constructed as follows. First, the corpus is processed document by document. For each document d, we collect the words appearing in d and, for every word w, we store the number of times it appears (its "frequency"). Next, for every w in d, we obtain the set of its *representative hypernyms*. We make no attempt at word-sense disambiguation and leave this aspect for further work. Instead, for words with several corresponding *synsets* (words with multiple senses), a hypernym h is representative if it is a hypernym for every sense of the word w, by which we avoid the fact that we are missing information on the actual sense of the word in context. Thus, we identify the set of all corresponding WordNet synsets of w (denoted by $\mathfrak{S}_w$), and the representative hypernyms of word w, denoted by $\mathfrak{A}_w$, are hypernyms of all the synonyms in $\mathfrak{S}_w$:

$$\mathfrak{A}_w = \bigcap_{s \in \mathfrak{S}_w} \{h | h \text{ is a hypernym of } s\}.$$

We also store "frequencies" of all representative hypernym counts—for a hypernym h, the frequency of h is defined as the sum of the frequencies of all of its hyponyms. Note that more general

hypernyms will occur more often, hence the hierarchical relations between hypernyms are captured via hypernym frequency.

Once representative hypernyms are identified for all words appearing in a document d, the set $\mathfrak{H}_d$ is constructed as $\mathfrak{H}_d = \bigcup_{w \in d} \mathfrak{A}_w$, and, once this set is constructed for all documents, the set $\mathfrak{H}$ is constructed as the set of all representative hypernyms of the corpus, i.e., $\mathfrak{H} = \bigcup_{d \in \mathcal{D}} \mathfrak{H}_d$. Throughout this process, counts of hypernym occurences are stored for each document, and once all documents are processed, features are constructed based on the overall hypernym counts. The number of semantic feature vectors to be constructed, denoted λ, is a parameter of the proposed algorithm. The upper bound for λ is $|\mathfrak{H}|$, i.e., the number of all representative hypernyms. We propose three approaches, which prioritize the hypernyms according to their frequency of occurrence. The three approaches used to select λ hypernyms for semantic feature vector construction are:

- top λ most frequent terms,
- last λ terms (very rare terms),
- a set of random λ terms.

The obtained matrix can be used for learning either as a separate semantic feature set (S) or as the whole DS matrix along with word-index matrix D.

3.2. Learning from the Semantic Space

The second step of the SRNA approach consists of training a deep architecture using the expanded feature matrix (DS) obtained in the first step. In SRNA, semantic features are fed into a deep architecture along with document vectors. The outline of the architecture, shown in Figure 2, can be represented in three main parts. The first part is responsible for learning from document vectors, and is denoted by $\mathfrak{D}$. The second part learns from the constructed semantic vectors, denoted as $\mathfrak{S}$. Finally, before output layer, outputs of $\mathfrak{D}$ and $\mathfrak{S}$ are merged and processed jointly. We denote this part by $(\mathfrak{D} + \mathfrak{S})$. We give exact (hyperparameter) parameterization of the architecture in Section 4.

The recurrent part of the network, represented by the $\mathfrak{D}$ part, is in this work defined as follows. An input vector of word indices is first fed into an embedding layer with dropout regularization. The resulting output is used in a standard LSTM layer. The output of this step is activated by a ReLU activation function, defined as:

$$ReLU(x) = max(0, x).$$

The output of this layer is followed by a MaxPooling layer. Here, maximal values of a kernel moving across the input vector are extracted. Finally, a dense layer with dropout regularization is used. Formally, the $\mathfrak{D}$ part of the network can be defined as:

$$\begin{aligned} L_{(1)} &= Dropout(Emb(D)), \\ L_{(2)} &= MaxPooling(ReLU_{(2)}(LSTM(L_{(1)}))), \\ L_{(3w)} &= Dropout(W_{(3)}^T L_{(2)} + b_{(3)}). \end{aligned}$$

The $\mathfrak{S}$ part of the architecture similarly consists of fully connected layers. The input for this part of the network are generated semantic features S. It can be represented as:

$$\begin{aligned} L_{(1)} &= Elu_{(1)}(W_{(1)}^T S + b_{(1)}), \\ L_{(2)} &= Dropout(L_{(1)}), \\ L_{(3s)} &= Elu_{(3)}(W_{(3)}^T L_{(2)} + b_{(3)}). \end{aligned}$$

Here, we use the exponential linear unit [43], defined as

$$Elu(x) = \begin{cases} x, & \text{for } x \geq 0, \\ c(e^x - 1), & \text{for } x < 0. \end{cases}$$

Here, c is a constant determined during parameterization of the architecture. Outputs of D and S parts of the architecture are concatenated and used as input to a set of fully connected (dense) layers (M), defined as:

$$\begin{aligned} L_{(1)} &= concat(L_{(3w)}, L_{(3s)}), \\ L_{(2)} &= Elu(Dropout(W_{(2)}^T L_{(1)} + b_{(2)})), \\ L_{(3f)} &= \sigma(W_{(3)}^T L_{(2)} + b_{(3)}). \end{aligned}$$

The *concat* operator merges the outputs of the two individual parts of the network into a single matrix. For concatenation, one of the dimensions (in our case, N, the number of instances) of the two output layers must be the same.

Finally, the output layer $L_{(3f)}$ includes one neuron for each class in the data set. We use binary cross entropy as the loss function. The exact layer parameterizations are discussed in the experimental setting section. The Adam optimizer [44] was chosen due to faster convergence. Formulation of the whole SRNA approach is presented in Algorithm 1.

Algorithm 1 Semantic space propositionalization with learning.

```
Data: corpus 𝒟,WordNet taxonomy
for all document in 𝒟 do
    for all word in document do
        Find hypernyms (based on WordNet) for word, store them and their counts
    end for
    Compute intersection of hypernym paths
end for
Assign feature values based on hypernym frequency in a document
S := Select top λ hypernyms as features based on overall hypernym frequency
D := transform 𝒟 into a matrix of word indices Learn a deep model using matrices D and S.
```

The proposed algorithm's temporal complexity is linear with respect to document number, making it scalable even for larger corpora. Similarly, the frequency count estimation is not computationally expensive. One of the key goals of this work was to explore how semantic information, derived from individual documents, affects the learner's performance. The SRNA code is accessible at https://gitlab.com/skblaz/srna.

In the next section, we continue with the experimental setting where we evaluate the proposed methodology.

4. Experimental Setting

We compared the performance of the SRNA approach against multiple baseline classifiers. We tested the methods on three benchmark data sets. We next describe the experimental setting in more detail.

4.1. Data Sets

All documents were padded to the maximum dimension of 150 words. We conduct a series of experiments, where we truncate the training documents (D) to lengths from 15 to 150 by the increment of 10. The semantic feature matrix S is constructed using truncated documents. Note that

the number of documents remains the same; we only experiment with the number of words per document. The results were obtained using 10 fold stratified cross validation. We tested the proposed approach on three data sets, listed below.

Reuters data set consists of 11,263 newspaper articles, belonging to 46 different topics (classes). This data set is loaded via the Keras library, where it is also publicly accessible (https://keras.io/datasets/).

IMDB review data set consists of 50,000 reviews. Here, the goal is to predict the sentiment of individual reviews (positive or negative). The data set was obtained from the Keras library [45], where it is also accessible.

PAN reviews data set consists of reviews written by 4160 authors (2080 male and 2080 female). Reviews written by the same author are concatenated in a single document. The goal is to classify the author's gender. Detailed description of the data set is given in [10].

4.2. Semantic Feature Construction

We generated 1000 semantic features for each of the feature selection approaches. After initial tests, we observed that the sparse feature set (rarest hypernyms) outperforms the other two approaches, thus this setting was used for further tests. To reduce the number of candidate hypernym features, we introduce a minimum frequency threshold—a threshold above which we consider a hypernym as a potential feature. The frequency threshold used was 10, i.e., a hypernym is common to at least 10 words from the corpus in order to be considered for feature construction. (Note that this step of the approach could be possibly improved using e.g., the RelieF) [46] branch of algorithms.

4.3. Deep Neural Architectures Used

As part of experimental evaluation, we test three deep learning models, two with inclusion of semantic vectors and a baseline ConvNet. All the models are initiated in the same way.

SRNA: Recurrent architecture. This is the proposed architecture that we described in Section 3. It learns by using LSTM cells on the sequential word indices, and simultaneously captures semantic meaning using dense layers over the semantic feature space.

Baseline RNN. The baseline RNN architecture consists of the non-semantic part of SRNA. Here, a simple unidirectional RNN is trained directly on the input texts.

Baseline CNN. The baseline neural networks used are a 1D convolutional neural network and a recurrent neural network with the same architecture as SRNA, where we omit the semantic part. Here, only word index vectors are used as inputs. The network was parameterized as follows. The number of filters was set to 64, the kernel size used was 5. The MaxPooling region was of size 5. The outputs of the pooling region were used as input to a dense layer with 48 neurons, followed by the final layer.

One of the main problems with small data sets and neural networks is overfitting. Each neural network is trained incrementally, where the training is stopped as soon as the network's performance starts to degrade. Furthermore, dropout layers are used for additional regularization (the dropout rate was set to 0.5). The alpha parameter of each *Elu* activation function was set to 1.

As an additional baseline, we implemented also two non-neural classifiers, i.e., the random forest classifier, and a support vector machine, where we also tested how semantic vectors contribute to classification accuracy.

The random forest (RF) classifier was initialized as follows: number of trees for classification from documents was set to the average document length present in a given corpus rounded to the closest integer. One versus all (OVA) classification scheme was used for the multi-class Reuters task. To evaluate the semantic addition, we implemented two variants of random forests, both learned from identical input as given to neural networks. **Semantic RF** is the random forest that

leverages semantic information (i.e., $D + S$ matrix), while **RF** is trained exclusively on TF-IDF word vectors obtained from D.

Support vector machine (SVM) classifier [47] was trained as follows. We used the RBF kernel and the C value determined over a grid search over range [0.1,1,10]. Similarly to random forests, we also implemented the version called **Semantic SVM**, which uses SRNA's semantic features along with TF-iDF matrix as input.

Other Technical Details

The SRNA approach was along with Baseline RNN and CNN architectures implemented in Keras framework, where we used the Tensorflow computational back-end [48]. The other classifiers were called from the Scikit-learn Python library [49]. All approaches were tested on a Nvidia Titan GPU (NVIDIA, Santa Clara, CA, USA). The baseline Random Forest classifier was implemented in Scikit-learn [49]. Matrix-based operations in the propositionalization step used the Numpy library [50].

5. Results and Discussion

For all data sets, we measure the accuracy. In case of Reuters, which is a multiclass problem, the exact accuracy is also termed subset accuracy (or exact match ratio). We also compute the F1 score for the IMDB and PAN data sets, and micro F1 for Reuters. Each experiment with 10 fold cross validation is repeated five times, and the results are averaged. To statistically evaluate the results, we used the Friedman's test, followed by the Nemenyi post hoc correction. The results are presented according to the classifier's average ranks along a horizontal line [51]. The obtained critical distance diagrams are interpreted as follows: if one or more classifiers are connected with a bold line, their performance does not differ significantly (at alpha = 0.05). We rank the classifiers for each data set, for each individual subsample. Furthermore, we visualize the performance of SRNA compared to baseline RNN using the recently introduced Bayesian hierarchical t-test—a Bayesian alternative to pairwise classifier comparison over multiple data sets [52]. Here, instead of significance level, a rope parameter is set. This parameter determines the threshold, under which we consider the difference in classifier performance to be the same. In this work, we set this threshold to 0.01. Note that the hierarchical Bayesian t-test offers the opportunity to explore the pairwise comparison of classifiers in more detail, hence we use it to inspect the SRNA vs. Baseline RNN combination.

For different document lengths, we calculate the accuracy and F1 scores, for which the plots (for the sequence length up to 100) are provided in Figures 3 and 4, respectively. It can be seen that, on the Reuters data set, SRNA outperforms other approaches in terms of Accuracy and F1, while for the other two data sets it achieves comparable results to baseline RNN and CNN.

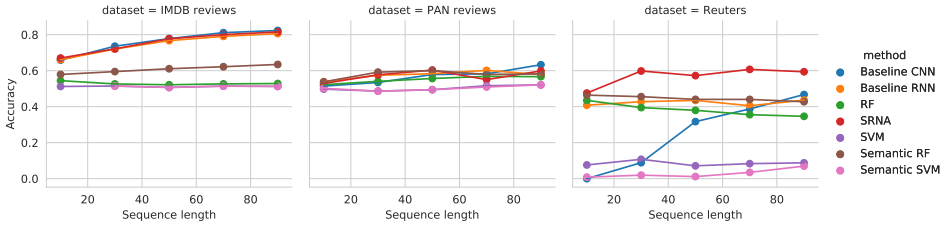


Figure 3. Accuracy results on three benchmark data sets.

We also present critical distance diagrams for the accuracy (Figure 5) and F1 measures (Figure 6). From the ranks, we can see that the SRNA approach outperforms all other baselines. However, the differences in performance between the SRNA approach and Baseline RNNs (as well as most of other classifiers) are not significant, and are data set dependent.

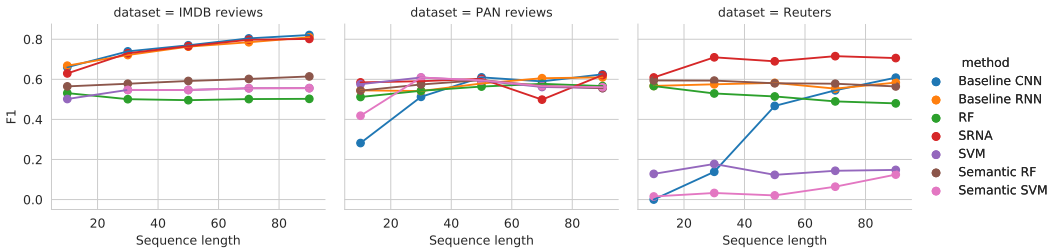


Figure 4. F1 results on three benchmark data sets.

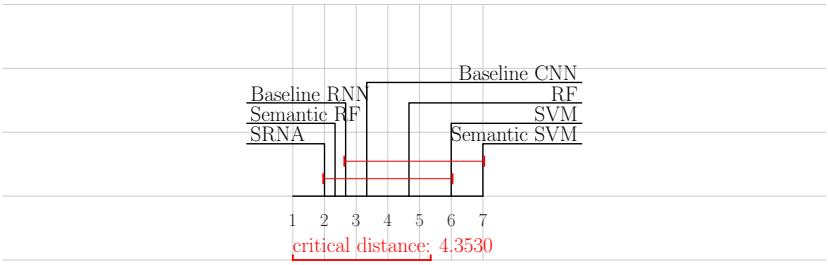


Figure 5. Accuracy—CD diagram.

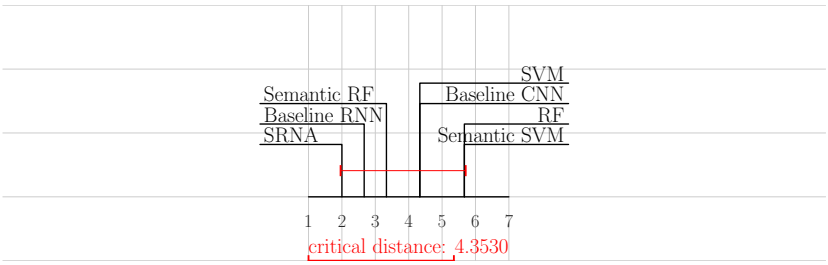


Figure 6. (Micro) F1—CD diagram.

Interestingly, the semantic feature-augmented random forests on average outperform their basic counterparts. This observation indicates that the semantic features could be used in a general classification setting, where an arbitrary classifier could benefit from the background knowledge introduced. Rigorous, large-scale experimental proof of this statement is out of the scope of this study.

As the goal of the proposed SRNA approach is to improve learning on very small data sets, with very limited data, we further investigate the classifier's performance on up to 100 words (see Figures 3 and 4).

When the considered recurrent architectures were inspected in more detail (Figure 7), we can observe that there is a higher probability that SRNA outperforms (Prob = 0.64) the baseline RNNs (Prob = 0.30), when the region of practical equivalence (ROPE) is set to 0.01, even though the performances of the two architectures are very similar. As an input to this test, we used differences in classifiers' performances from five repetitions of 10 fold cross validation.

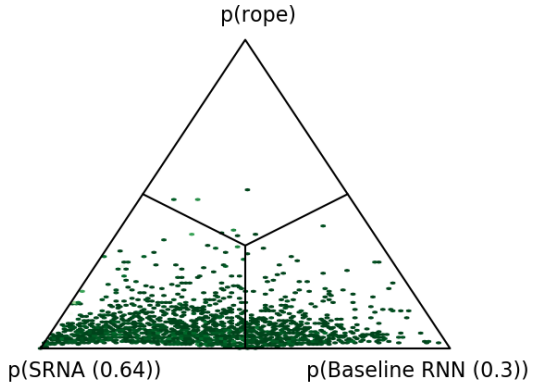


Figure 7. Sampled probability density of differences in classifier performance. Overall, the SRNA approach outperforms the baseline RNN, yet the larger differences in performance (e.g., Reuters data set) are data set-dependent. Higher probability of winning (0.64) in favour of SRNA indicates that semantic features potentially improve performance. Note that the ROPE parameter was for this test set to 0.01.

We further investigate the reasons why the baseline convolutional network performs very poorly when only up to 50 words are used. We believe the poor performance is related to the small data size. The CNN learns normally on very reduced documents, yet when its predictions were inspected, we observed it was not able to produce a single positive classification.

This behaviour was observed for document length ≤ 50, which resulted in two valid classifications ($length = 50$), whereas all other classifications ($length < 50$) returned $\sim$0% accuracy. The difference in accuracy for very short document lengths serves as an additional empirical proof that semantic vectors can at least augment the signal up to the classification threshold when using the SRNA.

The SVM approaches do not perform well in the conducted experiments. We believe the reason for this could lie in too small grid search region, as well as the noise potential introduced by semantic features. This indicates that the semantic features could be further pruned—such noise can have observable effects on the network's performance when semantic vectors are merged with the word vectors.

In addition, we observe that the SVM classifier did not perform well also when semantic features were added. Even though we did not test the regularization (C) range exhaustively, we believe that the SVMs' performance could be further improved. Moreover, the RBF kernel is not necessarily the optimal kernel choice.

Furthermore, we discuss the performance of random forests. The random forest classifier is in the majority of settings outperformed by other approaches (apart from SVMs), which is not surprising as very simple forest construction was used. However, we can see that with random forests the use of semantic features provides improvement. As compared to SVMs, random forests use a relatively low number of features; it is therefore easier to observe a difference in performance when novel features are introduced.

Interestingly, the random forest's performance appears to degrade in the case of the Reuters data set, which could indicate overfitting. As we used an OVA classification scheme, this decline in performance could be possibly solved by more advanced multi-class approaches, such as some form of predictive clustering trees. It is also possible that the problem is simply too hard for a random forest classifier used in this study, as it was not able to recognize any meaningful pattern, useful for classification into one of the possible topics.

Even though this study is not devoted to improving the overall state-of-the-art classification performance (SOTA), but to demonstrate how semantic features contribute to their semantically

unaware counterparts, and especially how semantic features can be introduced in the neural architectures, we briefly discuss here SOTA results.

Currently, the best accuracy for the IMDB data set is estimated at around 98% for an approach that is based on paragraph vectors [53]. The authors compared their approach also with simple LSTMs (as used for baseline in this study), and obtained accuracies of 96%. We tested our baseline on the whole data set, and it performed similarly (95.3%), which serves as a validation of the baseline approach used in this study. Next, the accuracy on the Reuters data set was recently reported to be 80–85%, where multi-objective label encoders were used [54]. Our baseline implementation performs with 75% accuracy. Finally, SOTA for gender classification on PAN 2014 was reported to be around 73% [10].

Even if we investigated a particular aspect of text classification, not directly associated with SOTA, we will try to perform a more systematic evaluation to SOTA approaches in future work, however there are some limitations, such as computational cost of training very large networks and the fact that the majority of SOTA approaches do not account for a situation with sparse data. However, we believe that the proposed approach can be adapted to make current SOTA architectures more robust, especially when only fragments of inputs are considered.

6. Conclusions and Further Work

We developed an approach for propositionalization of semantic space in the form of taxonomies to improve text classification tasks. We explore possible deep architectures, which learn separately from the two feature spaces and prove that construction of such architectures can significantly improve overall classification on short document fragments. As we tested only three simple approaches for feature selection, this work could further benefit from more advanced feature selection techniques, such as the ones based on evolutionary computation or ReliefF branch of algorithms. We believe a more sophisticated feature selection approach would result in more relevant features, and could as such significantly speed up the learning phase. Furthermore, the approach could be tested in a setting where no feature selection is performed at all—for such experiments, one would need significantly more performant GPUs than the ones used in this experiment. We believe the neural networks would be able to select relevant features in an end-to-end manner.

As the results in this study indicate, recurrent neural architecture can indeed benefit from addition of semantic information, and part of the further work includes more extensive experimental tests, where state-of-the-art approaches, such as RMDL, HDLTex or hierarchical attention networks shall be combined with the proposed hypernym features.

As current state-of-the-art text classification approaches also work on the character level, it is yet to be investigated whether the proposed approach can also boost performance for character level architectures. Furthermore, the SRNA approach could potentially benefit from different types of recurrent layers, such as, for example, gated recurrent units (GRUs).

Last but not least, in a higher performance setting, the effects of semantic features could be evaluated on current SOTA algorithms, as well as on inherently short texts, such as tweets and comments. We will also include comparison of the proposed approach of semantic knowledge integration to enrichment with precomputed word embeddings.

Author Contributions: Conceptualization, B.Š., S.P. and J.K.; methodology, B.Š., J.K.; software, B.Š.; validation, B.Š., J.K., N.L. and S.P.; formal analysis, J.K.; investigation, S.P., B.Š.; resources, B.Š., S.P.; data curation, B.Š.; writing—original draft preparation, all authors; writing—review and editing, all authors; visualization, B.Š.; supervision, N.L., S.P.; project administration, S.P., N.L.; funding acquisition, S.P., N.L.

Funding: The work of the first author was funded by the Slovenian Research Agency through a young researcher grant. The work of other authors was supported by the Slovenian Research Agency (ARRS) core research programme *Knowledge Technologies* (P2-0103) and ARRS funded research project *Semantic Data Mining for Linked Open Data* (financed under the ERC Complementary Scheme, N2-0078). This paper is supported also by the European Union's Horizon 2020 research and innovation programme under Grant No. 825153, EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media). The results of this

publication reflect only the authors' views and the Commission is not responsible for any use that may be made of the information it contains.

Acknowledgments: The GPU used for this research was donated by the NVIDIA Corporation.

Conflicts of Interest: The authors declare no conflict of interest.

References

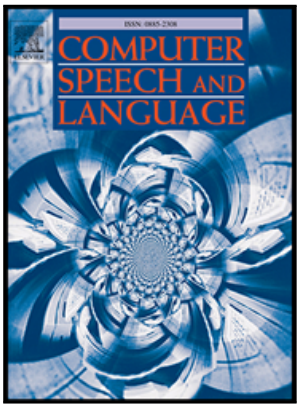
1. Aggarwal, C.C.; Zhai, C. A survey of text classification algorithms. In *Mining Text Data*; Springer: Boston, MA, USA, 2012; pp. 163–222.
2. Sebastiani, F. Machine learning in automated text categorization. *ACM Comput. Surv.* **2002**, *34*, 1–47. [CrossRef]
3. Tang, D.; Qin, B.; Liu, T. Document modeling with gated recurrent neural network for sentiment classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal, 17–21 September 2015; pp. 1422–1432.
4. Kusner, M.; Sun, Y.; Kolkin, N.; Weinberger, K. From word embeddings to document distances. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 957–966.
5. Ławrynowicz, A. *Semantic Data Mining: An Ontology-Based Approach*; IOS Press: Amsterdam, The Netherlands, 2017; Volume 29.
6. Vavpetič, A.; Lavrač, N. Semantic subgroup discovery systems and workflows in the SDM toolkit. *Comput. J.* **2013**, *56*, 304–320. [CrossRef]
7. Adhikari, P.R.; Vavpetič, A.; Kralj, J.; Lavrač, N.; Hollmén, J. Explaining mixture models through semantic pattern mining and banded matrix visualization. *Mach. Learn.* **2016**, *105*, 3–39. [CrossRef]
8. Scott, S.; Matwin, S. Text classification using WordNet hypernyms. In Proceedings of the Workshop on Usage of WordNet in Natural Language Processing Systems, Montreal, QC, Canada, 16 August 1998; University of Montreal: Montreal, QC, Canada, 1998; pp. 45–51.
9. Mansuy, T.N.; Hilderman, R.J. Evaluating WordNet features in text classification models. In Proceedings of the FLAIRS Conference, Melbourne Beach, FL, USA, 11–13 May 2006; American Association for Artificial Intelligence: Menlo Park, CA, USA, 2006; pp. 568–573.
10. Rangel, F.; Rosso, P.; Chugur, I.; Potthast, M.; Trenkmann, M.; Stein, B.; Verhoeven, B.; Daelemans, W. Overview of the 2nd author profiling task at PAN 2014. In Proceedings of the Working Notes Papers of the CLEF Conference, Sheffield, UK, 15–18 September 2014; pp. 1–30.
11. Rangel, F.; Rosso, P.; Verhoeven, B.; Daelemans, W.; Potthast, M.; Stein, B. Overview of the 4th author profiling task at PAN 2016: Cross-genre evaluations. In Proceedings of the Working Notes Papers of the CLEF Conference, Evora, Portugal, 5–8 September 2016; pp. 750–784.
12. Cho, J.; Lee, K.; Shin, E.; Choy, G.; Do, S. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *arXiv* **2015**, arXiv:1511.06348.
13. Landauer, T.K. *Latent Semantic Analysis*; Wiley Online Library: Hoboken, NJ, USA, 2006.
14. Blei, D.M.; Ng, A.Y.; Jordan, M.I. Latent dirichlet allocation. *J. Mach. Learn. Res.* **2003**, *3*, 993–1022.
15. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781.
16. 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), Doha, Qatar, 25–29 October 2014; pp. 1532–1543.
17. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching Word Vectors with Subword Information. *arXiv* **2016**, arXiv:1607.04606.
18. Song, G.; Ye, Y.; Du, X.; Huang, X.; Bie, S. Short text classification: A survey. *J. Multimed.* **2014**, *9*, 635–644. [CrossRef]
19. Tang, D.; Wei, F.; Yang, N.; Zhou, M.; Liu, T.; Qin, B. Learning sentiment-specific word embedding for twitter sentiment classification. In Proceedings of the 52nd ACL Conference, Baltimore, MD, USA, 23–25 June 2014; Volume 1, pp. 1555–1565.
20. Cagliero, L.; Garza, P. Improving classification models with taxonomy information. *Data Knowl. Eng.* **2013**, *86*, 85–101. [CrossRef]

21. Škrlj, B.; Kralj, J.; Lavrač, N. CBSSD: Community-based semantic subgroup discovery. *J. Intell. Inf. Syst.* **2019**, 1–40. [CrossRef]
22. Xu, N.; Wang, J.; Qi, G.; Huang, T.S.; Lin, W. Ontological random forests for image classification. In *Computer Vision: Concepts, Methodologies, Tools, and Applications*; IGI Global: Hershey, PA, USA, 2018; pp. 784–799.
23. Elhadad, M.K.; Badran, K.M.; Salama, G.I. A novel approach for ontology-based feature vector generation for web text document classification. *Int. J. Softw. Innov.* **2018**, *6*, 1–10. [CrossRef]
24. Kaur, R.; Kumar, M. Domain ontology graph approach using Markov clustering algorithm for text classification. In Proceedings of the International Conference on Intelligent Computing and Applications, Madurai, India, 14–15 June 2018; Springer: Berlin, Germany, 2018; pp. 515–531.
25. Ristoski, P.; Faralli, S.; Ponzetto, S.P.; Paulheim, H. Large-scale taxonomy induction using entity and word embeddings. In Proceedings of the International Conference on Web Intelligence, Leipzig, Germany, 23–26 August 2017; pp. 81–87.
26. Liu, Q.; Jiang, H.; Wei, S.; Ling, Z.H.; Hu, Y. Learning semantic word embeddings based on ordinal knowledge constraints. In Proceedings of the 53rd ACL Conference and the 7th IJCNLP Conference, Beijing, China, 26–31 July 2015; Volume 1, pp. 1501–1511.
27. Bian, J.; Gao, B.; Liu, T.Y. Knowledge-powered deep learning for word embedding. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Nancy, France, 15–19 September 2014; Springer: Berlin, Germany, 2014; pp. 132–148.
28. Zhang, X.; Zhao, J.; LeCun, Y. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*; Curran Associates, Inc.: Red Hook, NY, USA, 2015; pp. 649–657.
29. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436. [CrossRef]
30. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
31. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
32. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*; Curran Associates, Inc.: Red Hook, NY, USA, 2012; pp. 1097–1105.
33. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; Volume 1,
34. Gal, Y.; Ghahramani, Z. A theoretically grounded application of dropout in recurrent neural networks. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*; Curran Associates, Inc.: Red Hook, NY, USA, 2016; pp. 1019–1027.
35. Cheng, J.; Dong, L.; Lapata, M. Long short-term memory-networks for machine reading. *arXiv* **2016**, arXiv:1601.06733.
36. Graves, A.; Mohamed, A.R.; Hinton, G. Speech recognition with deep recurrent neural networks. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, BC, Canada, 26–31 May 2013; pp. 6645–6649.
37. Chung, J.; Gulcehre, C.; Cho, K.; Bengio, Y. Gated feedback recurrent neural networks. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 2067–2075.
38. Kowsari, K.; Heidarysafa, M.; Brown, D.E.; Meimandi, K.J.; Barnes, L.E. Rmdl: Random multimodel deep learning for classification. In Proceedings of the 2nd International Conference on Information System and Data Mining, Lakeland, FL, USA, 9–11 April 2018; pp. 19–28.
39. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
40. Kowsari, K.; Brown, D.E.; Heidarysafa, M.; Meimandi, K.J.; Gerber, M.S.; Barnes, L.E. Hdltex: Hierarchical deep learning for text classification. In Proceedings of the 16th IEEE International Conference on Machine Learning and Applications (ICMLA), Cancun, Mexico, 18–21 December 2017; pp. 364–371.
41. Cheng, H.T.; Koc, L.; Harmsen, J.; Shaked, T.; Chandra, T.; Aradhye, H.; Anderson, G.; Corrado, G.; Chai, W.; Ispir, M.; et al. Wide & deep learning for recommender systems. In Proceedings of the 1st Workshop on Deep Learning for Recommender Systems, Boston, MA, USA, 15 September 2016; pp. 7–10.
42. Miller, G.A. WordNet: A lexical database for English. *Commun. ACM* **1995**, *38*, 39–41. [CrossRef]

43. Clevert, D.A.; Unterthiner, T.; Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). *arXiv* **2015**, arXiv:1511.07289.
44. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
45. Chollet, F. Keras. 2015. Available online: https://github.com/fchollet/keras (accessed on 20 March 2019).
46. Robnik-Šikonja, M.; Kononenko, I. Theoretical and empirical analysis of ReliefF and RReliefF. *Mach. Learn.* **2003**, *53*, 23–69. [CrossRef]
47. Chang, C.C.; Lin, C.J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2011**, *2*, 27. [CrossRef]
48. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M.; et al. Tensorflow: A system for large-scale machine learning. In Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
49. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
50. Walt, S.V.D.; Colbert, S.C.; Varoquaux, G. The NumPy array: A structure for efficient numerical computation. *Comput. Sci. Eng.* **2011**, *13*, 22–30. [CrossRef]
51. Demšar, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **2006**, *7*, 1–30.
52. Benavoli, A.; Corani, G.; Demšar, J.; Zaffalon, M. Time for a change: A tutorial for comparing multiple classifiers through Bayesian analysis. *J. Mach. Learn. Res.* **2017**, *18*, 2653–2688.
53. Hong, J.; Fang, M. *Sentiment Analysis with Deeply Learned Distributed Representations of Variable Length Texts*; Technical Report; Stanford University: Stanford, CA, USA, 2015.
54. Zhang, H.; Xiao, L.; Chen, W.; Wang, Y.; Jin, Y. Multi-task label embedding for text classification. *arXiv* **2017**, arXiv:1710.07210.

Appendix E: tax2vec: Constructing Interpretable Features from Taxonomies for Short Text Classification

Journal Pre-proof



tax2vec: Constructing Interpretable Features from Taxonomies for Short Text Classification

Blaž Škrlj, Matej Martinc, Jan Kralj, Nada Lavrač, Senja Pollak

PII: S0885-2308(20)30037-1
DOI: https://doi.org/10.1016/j.csl.2020.101104
Reference: YCSLA 101104

To appear in: *Computer Speech & Language*

Received date: 31 January 2019
Revised date: 19 April 2020
Accepted date: 21 April 2020

Please cite this article as: Blaž Škrlj, Matej Martinc, Jan Kralj, Nada Lavrač, Senja Pollak, tax2vec: Constructing Interpretable Features from Taxonomies for Short Text Classification, *Computer Speech & Language* (2020), doi: https://doi.org/10.1016/j.csl.2020.101104

tax2vec: Constructing Interpretable Features from Taxonomies for Short Text Classification

Blaž Škrlj[b,a], Matej Martinc[b,a], Jan Kralj[a], Nada Lavrač[a,c], *Senja Pollak[a]

[a] *Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia*
[b] *Jožef Stefan International Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia*
[c] *University of Nova Gorica, Glavni trg 8, 5271 Vipava, Slovenia*

Abstract

The use of background knowledge is largely unexploited in text classification tasks. This paper explores word taxonomies as means for constructing new semantic features, which may improve the performance and robustness of the learned classifiers. We propose tax2vec, a parallel algorithm for constructing taxonomy-based features, and demonstrate its use on six short text classification problems: prediction of gender, personality type, age, news topics, drug side effects and drug effectiveness. The constructed semantic features, in combination with fast linear classifiers, tested against strong baselines such as hierarchical attention neural networks, achieves comparable classification results on short text documents. The algorithm's performance is also tested in a few-shot learning setting, indicating that the inclusion of semantic features can improve the performance in data-scarce situations. The tax2vec capability to extract corpus-specific semantic keywords is also demonstrated. Finally, we investigate the semantic space of potential features, where we observe a similarity with the well known Zipf's law.

Keywords: taxonomies, vectorization, text classification, short documents, feature construction, semantic enrichment

*Corresponding author

1. Introduction

In text mining, document classification refers to the task of classifying a given text document into one or more categories based on its content [1]. Given an input set of labeled text documents, a text classifier is expected to learn to associate the patterns appearing in the documents to the document labels. Deep learning approaches [2] have recently become a standard in natural language-related learning tasks, demonstrating good performance on a variety of different classification tasks, including sentiment analysis of tweets [3] and news categorization [4]. Despite achieving state-of-the-art performance on many tasks, deep learning is not yet optimized for situations, where the number of documents in the training set is low or when the documents contain very little text [5].

Semantic data mining denotes a data mining approach where domain ontologies are used as background knowledge in the data mining process [6]. Semantic data mining approaches have been successfully applied to association rule learning [7], semantic subgroup discovery [8, 9], data visualization [10] and text classification [11]. Provision of semantic information allows the learner to use features on a higher semantic level, possibly enabling better data generalizations. The semantic information is commonly represented as relational data in the form of taxonomies or 3 ontologies. Development of approaches that leverage such information remains a lively research topic in several fields, including biology [12, 13], sociology [14] and natural language processing [15].

This paper contributes to semantic data mining by using word taxonomies as means for semantic enrichment by constructing new features, with the goal to improve the performance and robustness of the learned classifiers. In particular, it addresses classification of short or incomplete documents, which is useful in a large variety of text classification tasks. Short text is characterized by shortness in the text length, and sparsity in the terms presented, which results in the difficulty in managing and analyzing them based on the bag-of-words representation only. Short texts can be found everywhere, such as search snippets, product reviews and similar [16]. For example, in author profiling, the task is

to recognize the author's characteristics such as age or gender [17], based on a collection of author's text samples. Here, the effect of data size is known to be an important factor, influencing classification performance [18]. A frequent text type for this task are tweets, where a collection of tweets from the same author is considered a single document, to which a label must be assigned. The fewer instances (tweets) per user we need, the more powerful and useful the approach. Learning from only a handful of tweets can lead to preliminary detection of bots in social networks, and is hence of practical importance [19, 20]. In a similar way, this holds true for nearly any kind of text classification task. For example, for classifying news into a specific topic, using only snippets or titles may be preferred due to non-availability of entire news texts or for increasing the processing speed. Moreover, in biomedical applications, Grässer et al. [21] tried to predict drug's side effects and effectiveness from patients' short commentaries, while Boyce et al. [22] investigated the use of short user comments to assess drug-drug interactions.

It has been demonstrated that deep neural networks in general need a large amount of information in order to learn complex classifiers, i.e. they require a large training set of documents. For example, the recently introduced BERT neural network architecture [2] consisting of many hidden layers was trained on the whole Wikipedia. It was also shown that the state-of-the-art models do not perform well when incomplete (or scarce) information is used as input [23]. On the other hand, promising results regarding zero-shot [24] and few-shot [25] learning were recently achieved.

This paper proposes a novel approach named *tax2vec*, where semantic information available in taxonomies is used to construct semantic features that can improve classification performance on short texts. In the proposed approach, features are constructed automatically and remain *interpretable*. We believe that tax2vec could help explore and understand how external semantic information can be incorporated into existing (black-box) machine learning models, as well as help to explain what is being learned.

This work is structured as follows. Following the theoretical preliminaries

and the related work necessary to understand how semantic background knowledge can be used in learning, presented in Section 2, we continue with the description of the proposed tax2vec methodology in Section 3. In Section 4, we describe the experimental setting used to test the methodology. In Section 5, we present the results of experiments, including the evaluation of the qualitative properties of features constructed using tax2vec, and extensive classification benchmark tests. Section 6 discusses the properties of the resulting semantic space and the explainability of the proposed tax2vec algorithm. Implementation and availability of tax2vec is addressed in Section 7. The paper concludes with a summary and prospects for further work in Section 8. For completeness, Appendix A includes a detailed description of the Personalized PageRank algorithm, while Appendix B presents an example segmentation of news articles into paragraphs, forming short documents of interest for this study. Finally, Appendix C contains an additional ablation study regarding the impact of feature numbers on the classifier performance.

2. Background and related work

In this section we present the theoretical preliminaries and some related work, which served as the basis for the proposed tax2vec approach. We begin by explaining different levels of semantic context and the rationale behind the proposed approach.

2.1. Semantic context

Document classification is highly dependent on *document representation*. In simple bag-of-words representations, the frequency (or a similar weight such as term frequency-inverse document frequency—tf-idf) of each word or n-gram is considered as a separate feature. More advanced representations group words with similar meaning together. Such approaches include Latent Semantic Analysis [26], Latent Dirichlet Allocation [27], and more recently word embeddings [28]. It has been previously demonstrated that context-aware algorithms signifi-

cantly outperform the naive learning approaches [29]. We refer to such semantic context as the *first-level context*.

Second-level context can be introduced by incorporating *background knowledge* (e.g., ontologies) into a learning task, which can lead to improved interpretability and performance of classifiers, learned e.g., by rule learning [8] or random forests [30]. In text mining, Elhadad *et al.* [31] present an ontology-based web document classifier, while Kaur *et al.* [32] propose a clustering-based algorithm for document classification that also benefits from knowledge stored in the underlying ontologies. Cagliero and Garza [29] present a custom classification algorithm that can leverage taxonomies and demonstrate on a case study of geospatial data that such information can be used to improve the learner's classification performance. Use of hypernym-based features for classification tasks has been considered previously. For example, hypernym-based features were used in rule learning by the Ripper rule learning algorithm [11]. Moreover, it was also demonstrated that the use of hypernym-based features constructed from WordNet significantly impacts the classifier performance [33].

2.2. Feature construction and selection

When unstructured data is used as input, it is common to explore the options of feature construction. Even though recently introduced deep neural network based approaches operate on simple word indices (or byte-pair encoded tokens) and thus eliminate the need for manual construction of features, such alternatives are not necessarily the optimal approach when vectorizing the background knowledge in the form of taxonomies or ontologies. Features obtained by training a neural network are inherently non-symbolic and as such do not present any added value to the developer's understanding of the (possible) causal mechanisms underlying the learned classifier [34, 35]. In contrast, understanding the semantic background of a classifier's decision can shed light on previously not observed second-level context vital to the success of learning, rendering otherwise incomprehensible models easier to understand.

Definition 1 (Feature construction)**.** *Given an unstructured input consisting*

of n documents, a feature construction algorithm outputs a matrix $F \in \mathbb{R}^{n \times \alpha}$, where α denotes the predefined number of features to be constructed.

In practical applications, features are constructed from various data sources, including texts [36], graphs [37, 38], audio recordings and similar data [39]. With the increasing computational power at one's disposal, automated feature construction methods are becoming prevalent. Here, the idea is that given some criterion, the feature constructor outputs a set of features selected according to the criterion. For example, the tf-idf feature construction algorithm, applied to a given document corpus, can automatically construct hundreds of thousands of n-gram features in a matter of minutes on an average of-the-shelf laptop.

Many approaches can thus output too many features to be processed in a reasonable time, and can introduce additional noise, which renders the task of learning even harder. To solve this problem, one of the known solutions is *feature selection.*

Definition 2 (Feature selection)**.** *Let $F \in \mathbb{R}^{n \times \alpha}$ represent the feature matrix (as defined above), obtained during automated feature construction. A feature selection algorithm transforms matrix F to a matrix $F' \in \mathbb{R}^{n \times d}$, where d represents the number of desired features after feature selection.*

Feature selection thus filters out the (unnecessary) features, with the aim of yielding a compact, information-rich representation of the unstructured input. There exist many approaches to feature selection. They can be based on the individual feature's information content, correlation, significance etc. [40]. Feature selection is, for example, relevant in biological data sets, where only a handful of the key gene markers are of interest, and can be identified by assessing the impact of individual features on the target space [41].

2.3. Learning from graphs and relational information

In this section we briefly discuss the works that influenced the development of the proposed approach. One of the most elegant ways to learn from graphs is by transforming them into propositional tables, which are a suitable input

for many down-stream learning algorithms. Recent attempts to vectorization of graphs include the node2vec [42] algorithm for constructing features from homogeneous networks; its extension metapath2vec [43] for heterogeneous networks; its symbolic version SGE [38]; the mol2vec [44] vectorization algorithm for molecular data; the struc2vec [45] graph vectorization algorithm based on homophily relations between nodes, and more. All these approaches (apart from SGE) are sub-symbolic, as the obtained vectorized information (embeddings) are not interpretable. Similarly, recently introduced graph-convolutional neural networks also yield local node embeddings, which take node feature vectors into account [46, 47].

In parallel to graph-based vectorization, approaches which tackle the problem of learning from relational databases have also been developed. Symbolic (interpretable) approaches for this vectorization task, known under the term propositionalization, include RSD [48], a rule-based algorithm which constructs relational features; and wordification [49], an approach for unfolding relational databases into bag-of-words representations. The approach, described in the following sections, relies on some of the key ideas initially introduced in the mentioned works on propositionalization, as taxonomies are inherently relational data structures.

3. The tax2vec approach

In this section we outline the proposed tax2vec approach. We begin with a general description of classification from short texts, followed by the key features of tax2vec, which offer solutions to some of the currently not well explored issues in text mining.

3.1. The rationale behind tax2vec

In general text classification tasks, deep learning approaches have outperformed other classifiers [2]. However, in classification tasks involving short documents (tweets, opinions, etc.), particularly where the number of instances is

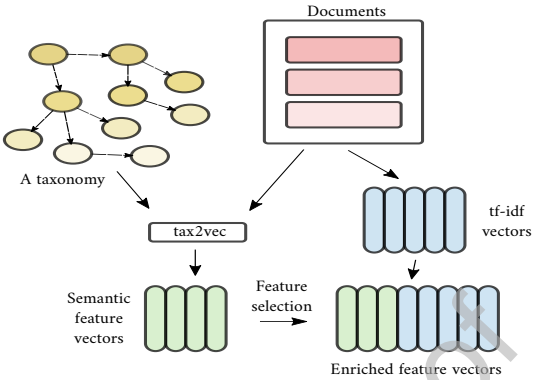


Figure 1: Schematic representation of tax2vec, combined with standard tf-idf representation of documents. Note that darker nodes in the taxonomy represent more general terms.

low, deep learners are still outperformed by simpler classifiers, such as SVMs [50]. This observation was a motivation for the development of the tax2vec algorithm, proposed in this paper. Compared to non-symbolic node vectorization algorithms discussed in the previous section, tax2vec uses hypernyms as potential features directly and thus makes the process of feature construction and selection possible without the loss of classifier's *interpretability*.

We present the proposed tax2vec algorithm for semantic feature vector construction that can be used to enrich the feature vectors constructed by the established text processing methods such as tf-idf. The tax2vec algorithm takes as input a labeled or unlabeled corpus of n documents and a word taxonomy. It outputs a matrix of *semantic feature vectors* in which each row represents a semantics-based vector representation of one input document. Example use of tax2vec in a common language processing pipeline is shown in Figure 1. Note that the obtained semantic feature vectors serve as additional features in the final, vectorized representation of a given corpus.

Let us first explore how parts of the WordNet taxonomy [51] related to the training corpus can be used for the construction of novel features, as such background knowledge can be applied in virtually every English text-based learning

setting, as well as for many other languages [52].

3.2. Deriving semantic features

The tax2vec approach implements a two-step semantic feature construction process. First, a document-specific taxonomy is constructed, then a term-weighting scheme is used for feature construction.

3.2.1. Document-based and corpus-based taxonomy construction

In the first step of the tax2vec algorithm, a corpus-based taxonomy is constructed from the input document corpus. In this section we describe how the words from individual documents of a corpus are mapped to terms of the WordNet taxonomy to construct a *document-based taxonomy* by focusing on semantic structures, derived exclusively from the *hypernymy* relation between words. Individual document-based taxonomies are then merged into a joint *corpus-based taxonomy*.

When constructing a document-based taxonomy, each word is mapped to the hypernym WordNet taxonomy. This results in a tree-like structure, which spans from individual words to higher-order semantic concepts. For example, given the word monkey, one of its mappings in the WordNet hypernym taxonomy is the term *mammal*, which can be further mapped to e.g., *animal* etc., eventually reaching the most general term, i.e. *entity*.

In order to construct the mapping, the first problem to be solved is *word-sense disambiguation*. For example, the word *bank* has two different meanings, when considered in the following two sentences:

River bank was enforced. | National bank was robbed.

There are many approaches to word-sense disambiguation (WSD). We refer the reader to [53] for a detailed overview of the WSD methodology.

In tax2vec, we use Lesk [54], the gold standard WSD algorithm, to map each disambiguated word to the corresponding term in the WordNet taxonomy. The identified term is then associated with a path in the WordNet taxonomy

leading from the given term to the root of the taxonomy. Example hypernym path (with WordNet-style notation), extracted for word "astatine", is shown in Figure 2.

$Synset('entity.n.01')$
$\rightarrow Synset('abstraction.n.06')$
$\rightarrow Synset('relation.n.01')$
$\rightarrow Synset('part.n.01')$
$\rightarrow Synset('substance.n.01')$
$\rightarrow Synset('chemical\_element.n.01')$
$\rightarrow Synset('astatine.n.01')$

Figure 2: Example hypernym path extracted for word "astatine", where the $\rightarrow$ corresponds to the "hypernym of" relation (the majority of hypernym paths end with the "entity" term, as it represents one of the most general objects in the taxonomy).

By finding a hypernym path to the root of the taxonomy for all words in the input document, a *document-based taxonomy* is constructed, which consists of all hypernyms of all words in the document. After constructing the document-based taxonomy for all the documents in the corpus, the taxonomies are joined into a *corpus-based taxonomy*.

Note that processing each document and constructing the document-based taxonomy is entirely independent from other documents, allowing us to process the documents in parallel and join the results only when constructing the joint corpus-based taxonomy.

3.2.2. Semantic feature construction

During the construction of a document-based taxonomy, document-level term counts are calculated for each term. For each word t and document D, we count the number $f_{t,D}$ of times the word or one of its hypernyms appeared in a given document D.

The obtained counts can be used for feature construction directly: each term t from the corpus-based taxonomy is associated with a feature, and a document-

level term count is used as the feature value. The current implementation of tax2vec weights the feature values using the double normalization tf-idf metric. For term t, document D and user-selected normalization factor K, feature value tf-idf(t,D,K) is calculated as follows [55]:

$$\text{tf-idf}(t, D, K) = \underbrace{\left(K + (1-K)\frac{f_{t,D}}{\max_{\{t' \in D\}} f_{t',D}}\right)}_{\text{Weighted term frequency}} \cdot \underbrace{\log\left(\frac{N}{n_t}\right)}_{\substack{\text{Inverse} \\ \text{document frequency}}} \quad (1)$$

where $f_{t,D}$ is the term frequency, normalized by $\max_{\{t' \in D\}} f(t', D)$, which corresponds to the raw count of the most common hypernym of words in the document; value N represents the total number of documents in the corpus, n_t denotes the number of document-based taxonomies the hypernym appears in (i.e. the number of documents that contain a hyponym of t). Note that the term frequencies are normalized with respect to the most frequently occurring term to prevent a bias towards longer documents. In the experiments the normalization constant K was set to 0.5.

3.3. Feature selection

The problem with the above presented approach is that all hypernyms from the corpus-based taxonomy are considered, and therefore, the number of columns in the feature matrix can grow to tens of thousands of terms. Including all these terms in the learning process introduces unnecessary noise, and unnecessarily increases the spatial complexity. This leads to the need of feature selection (see Definition 2 in Section 2.2) to reduce the number of features to a user-defined number (a free parameter specified as part of the input). We next describe the scoring functions of feature selection approaches considered in this work.

As part of tax2vec, we implemented both supervised (Mutual Information - MI and Personalized PageRank - PPR), as well as unsupervised (Betweenness centrality - BC and term count-based selection) feature selection methods, discussed below. Note that the feature selection process is conducted *exclusively* on the semantic space (i.e. on the mapped WordNet terms).

Feature selection by term counts. Intuitively, the rarest terms are the most document-specific and could provide additional information to the classifier. This is addressed in tax2vec by the simplest heuristic, used in the algorithm: a term-count based heuristic that simply takes overall counts of all hypernyms in the corpus-based taxonomy, sorts them in ascending order according to their frequency of occurrence and takes the top d.

Feature selection using term betweenness centrality. As the constructed corpus-specific taxonomy is not necessarily the same as the WordNet taxonomy, the graph-theoretic properties of individual terms within the corpus-based taxonomy could provide a reasonable estimate of a term's importance. The proposed tax2vec implements the betweenness centrality (BC) [56] measure of individual terms as the scoring measure. The betweenness centrality is defined as:

$$BC(t) = \sum_{u \neq v \neq t} \frac{\sigma_{uv}(t)}{\sigma_{uv}}; \tag{2}$$

where σ_{uv} corresponds to the number of shortest paths (see Figure 3) between nodes u and v, and $\sigma_{uv}(t)$ corresponds to the number of paths that pass through term (node) t. Intuitively, betweenness measures the t's importance in the corpus-based taxonomy. Here, the terms are sorted in a descending order according to their betweenness centrality, and again, the top d terms are used for learning.

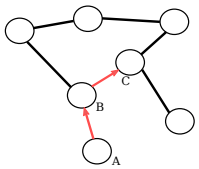


Figure 3: An example shortest path. The path colored red represents the smallest number of edges needed to reach node C from node A.

Feature selection using mutual information. The third heuristic, mutual information (MI) [57], aims to exploit the information from the labels,

assigned to the documents used for training. The MI between two random discrete variables represented as vectors F_i and Y (i.e. the i-th hypernym feature and a target binary class) is defined as:

$$MI(F_i, Y) = \sum_{x,y \in \{0,1\}} p(F_i = x, Y = y) \cdot \log_2 \left(\frac{p(F_i = x, Y = y)}{p(F_i = x) \cdot p(Y = y)} \right) \quad (3)$$

where $p(F_i = x)$ and $p(Y = y)$ correspond to marginal distributions of the joint probability distribution of F_i and Y. Note that for this step, tax2vec uses the binary feature representation, where the tf-idf features are rounded to the closest integer value (either 0 or 1). This way, only well represented features are taken into account. Further, tax2vec uses one-hot encodings of target classes, meaning that each target class vector consists exclusively of zeros and ones. For *each* of the target classes, tax2vec computes the mutual information (MI) between *all* hypernym features (i.e. matrix X) and a given class. Hence, for each target class, a vector of mutual information scores is obtained, corresponding to MI between individual hypernym features and a given target class.

Finally, tax2vec sums the MI scores obtained for each target class to obtain the final vector, which is then sorted in descending order. The first d hypernym features are used for learning. At this point tax2vec yields the selected features as a sparse matrix, maintaining the spatial complexity amounting to the number of float-valued non-zero entries.

Personalized PageRank-based hypernym ranking. Advances by Kralj *et al.* [58, 59] in learning using extensive background knowledge for rule induction explored the use of Personalized PageRank (PPR) algorithm for node subset selection in semantic search space exploration. In tax2vec, we use the same idea to prioritize (score) hypernyms in the corpus-based taxonomy. In this section, we first briefly describe the Personalized PageRank algorithm and then describe how it is applied in tax2vec.

The PPR algorithm takes as an input a network and a set of starting nodes in the network and returns a vector assigning a score to each node

in the input network. The scores of nodes are calculated as the stationary distribution of the positions of a random walker that starts its walk on one of the starting nodes and, in each step, either randomly jumps from a node to one of its neighbors (with probability p, set to 0.85 in our experiments) or jumps back to one of the starting nodes (with probability $1-p$). Detailed description of the PPR used in tax2vec is given in Appendix A. The PPR algorithm is used in tax2vec as follows:

1. Identify a set of hypernyms in the corpus-based taxonomy, to which the words in the input corpus map to in the first step of tax2vec (described in Section 3.2.1).
2. Run the PPR algorithm on the corpus-based taxonomy, using the hypernyms identified in step 1 as the starting set.
3. Use the top d best ranked hypernyms as candidate features.

Note that this heuristics offers *global* node ranks with respect to the corpus used.

3.4. The tax2vec algorithm

All the aforementioned steps form the basis of tax2vec, outlined in Algorithm 1. First, tax2vec iterates through the given labeled document corpus in parallel (lines 3–7). For each document, *MaptoTaxonomy* method identifies a set of disambiguated words and determines their corresponding terms in taxonomy $\mathfrak{T}$ (i.e. WordNet) using method m (i.e. Lesk). Term counts are stored for later use (*storeTermCounts*), and the taxonomy, derived from a given document (*doc*) is added to the corpus taxonomy $\mathfrak{T}_{\text{CORPUS}}$. Once traversed, the terms present in $\mathfrak{T}_{\text{CORPUS}}$ represent potential *features*. Term counts, stored for each document are aggregated into vectors of size n, where n is the number of documents in the corpus. The result of this step is a real-valued, sparse matrix (vecSpace), where columns represent all possible terms from $\mathfrak{T}_{\text{CORPUS}}$. In the following step, feature selection is conducted. Here, graph-based methods (e.g., BC and PPR) identify top d terms based on $\mathfrak{T}_{\text{CORPUS}}$'s properties (lines 9–12),

Algorithm 1: tax2vec

Data: Training set documents D, training document labels Y_{tr}, WordNet taxonomy $\mathfrak{T}$, word-to-taxonomy mapping m, feature selection heuristic h, number of selected features d

1 $\mathfrak{T}_{\text{CORPUS}} \leftarrow$ empty structure;
2 termCounts $\leftarrow$ empty structure;
3 **for** *doc* $\in D$ *(in parallel)* **do**
4 $\quad \mathfrak{T}_{\text{DOCUMENT}} \leftarrow$ MaptoTaxonomy($doc, \mathfrak{T}, m$);
5 $\quad$ Add storeTermCounts($\mathfrak{T}_{\text{DOCUMENT}}$) to termCounts;
6 $\quad$ Add $\mathfrak{T}_{\text{DOCUMENT}}$ to $\mathfrak{T}_{\text{CORPUS}}$;
7 **end**
8 vecSpace $\leftarrow$ tf-idf(constructTfVectors($D, \mathfrak{T}_{\text{CORPUS}}$,termCounts));
9 **if** *h is graph-based* **then**
10 $\quad$ topTerms $\leftarrow$ selectFeatures(h, $\mathfrak{T}_{\text{CORPUS}}$, d, optional Y_{tr});
11 $\quad$ selectedFeatures $\leftarrow$ select topTerms from vecSpace;
12 **end**
13 **else**
14 $\quad$ selectedFeatures$\leftarrow$ selectFeaturesDirectly(h, vecSpace,d ,Y_{tr});
15 **end**
16 **return** selectedFeatures;

Result: d new feature vectors in sparse vector format.

and non-graph methods (e.g., MI) is used directly on the sparse matrix to select which d features are the most relevant (lines 13–15). Finally, $selectedFeatures$, a matrix of selected semantic features is returned.

Note that in practice, tax2vec must also store the inverse document frequencies in order to generate features for unseen documents. We omit the description of this step for readability purposes.

3.5. Handling noise

Numerous data sets, including contemporary social media data sets, can be noisy and as such hard to handle by a learning system. We next discuss how distinct parts of tax2vec potentially handle noise in the data, including typos, incomplete and missing words and uncommon characters.

During the initial step of the semantic space construction, tax2vec conducts document-level word disambiguation in order to semantically characterize a given token (word). During this step, any tokens that are not present in the taxonomy will be ignored. Further, as word disambiguation requires a certain word window to operate, this hyperparameter can be used to control the size of context considered by tax2vec. In this work, however, we did not explicitly address the problem of invalid tokens in a given token's neighborhood, yet observed that small window sizes (two and three) offered reasonably robust performance.

Even though disambiguation with Lesk offers the initial *semantic pruning* capabilities, the tax2vec algorithm can further address potential noise as follows. As the user can determine the depth in the WordNet taxonomy that will be considered as the starting point for semantic space construction, potentially too specific terms can be avoided if necessary.

Finally, in the third step, tax2vec conducts *feature selection*. This part of the algorithm is responsible for *filtering* redundant and non-informative terms that could be considered as noise. We tested both supervised, as well as unsupervised feature selection methods, exploring whether additional information about class labels helps with term pruning. Apart from the semantic pruning and selection strategies discussed above, links, mentions and hashtags can be removed to further reduce the noise in social media texts (as mentioned in the description of the SVM implementation by Martinc et al. [60] in Section 4.2).

We believe all three steps to some extent address how noise is being handled. However, it is expected that additional grammar correction and text normalization could serve as a complementary step to offer improved performance on social media texts.

4. Experimental setting

This section presents the experimental setting used in testing the performance of tax2vec in document classification tasks. We begin by describing the data sets on which the method was tested. Next, we describe the classifiers used to assess the use of features constructed using tax2vec, along with the baseline approaches. We continue by describing the metrics used to assess classification performance, and the description of the experiments.

4.1. Data sets

We tested the effects of features produced with tax2vec on six different class labeled text data sets summarized in Table 1, intentionally chosen from different domains.

Table 1: Data sets used for experimental evaluation of tax2vec's impact on learning. Note that MNS corresponds to the maximum number of text segments (max. number of tweets or comments per user or number of news paragraphs as presented in Appendix B).

Data set (target)	Classes	Words	Unique words	Documents	MNS	Average tokens per segment
PAN 2017 (Gender)	2	5169966	607474	3600	102	14.23
MBTI (Personality)	16	11832937	372811	8676	89	27.98
PAN 2016 (Age)	5	943880	178450	402	202	13.17
BBC news	5	902036	58128	2225	76	70.39
Drugs (Side effects)	4	385746	27257	3107	3	41.47
Drugs (Overall effect)	4	385746	27257	3107	3	41.47

The first three data sets are composed of short documents from social media, where we consider classification of tweets.

PAN 2017 (Gender) data set. Given a set of tweets per user, the task is to predict the user's gender[1] [5].

MBTI (Meyers-Briggs personality type) data set. Given a set of tweets per user, the task is to predict to which personality class a user belongs[2], first discussed in [61].

[1] https://pan.webis.de/clef17/pan17-web

[2] https://www.kaggle.com/datasnaek/mbti-type/kernels

PAN 2016 (Age) data set. Given a set of tweets per user, the classifier should predict the users's age range[3] [18].

Next, we consider a news articles data set by which we test the potential of the method also on longer documents, while for few shot learning experiments (Section 5.3), we transform the setting to short text documents by using only few paragraphs per article and test whether competitive performance to full-text-based classification can be obtained.

BBC news data set. Given a news article (composed of a number of paragraphs)[4], the goal is to assign to it a topic from a list of topic categories[5] [62].

We also consider two biomedical data sets related to drug consumption. Here, the same training instances in the form of short user commentaries were used to predict two different targets.

Drug side effects. This data set links user opinions to side effects of a drug they are taking as treatment. The goal is to predict the side effects prior to experimental measurement [21].[6]

Drug effectiveness. Similarly to side effects (previous data set), the goal of this task is to predict drug effectiveness [21].

4.2. The classifiers used

As tax2vec serves as a preprocessing method for data enrichment with semantic features, arbitrary classifiers can use the resulting semantic features for learning. Note that in the experiments, the final feature space is composed of both semantic and non-semantic (original) features, i.e., the final feature set

[3] https://pan.webis.de/clef18/pan18-web
[4] Split to paragraphs according to the double new line is presented in Appendix B.
[5] https://github.com/suraj-deshmukh/BBC-Dataset-News-Classification/blob/master/dataset/dataset.csv
[6] http://archive.ics.uci.edu/ml/datasets

used for learning is formed *after* the semantic features have been constructed and selected, by concatenating the original features and the semantic features. We use the following learners:

PAN 2017 approach. An SVM-based approach that relies heavily on the method proposed by Martinc et al. [60] for the author profiling task in the PAN 2017 shared task [5]. This method is based on sophisticated hand-crafted features calculated on different levels of preprocessed text including optional social media text cleaning (e.g., Twitter hashtag, mentions, url replacement with filler tokens). The following features were used:

tf-idf weighted word unigrams calculated on lower-cased text with stopwords removed;

tf-idf weighted word bigrams calculated on lower-cased text with punctuation removed;

tf-idf weighted word bound character tetragrams calculated on lower-cased text;

tf-idf weighted punctuation trigrams (the so-called beg-punct [63], in which the first character is punctuation but other characters are not) calculated on lower-cased text;

tf-idf weighted suffix character tetragrams (the last four letters of every word that is at least four characters long [63]) calculated on lower-cased text;

emoji counts of the number of emojis in the document, counted by using the list of emojis created by [64][7]; this feature is only useful if the input text contains emojis;

document sentiment using the above-mentioned emoji list that contains the sentiment of a specific emoji, used to calculate the senti-

[7] http://kt.ijs.si/data/Emoji_sentiment_ranking/

ment of the entire document by simply adding the sentiment of all the emojis in the document; this feature is only useful if the input text contains emojis;

character flood counts calculated by the number of times that three or more identical character sequences appear in the document;

In contrast to the original approach proposed [60], we do not use POS tag sequences as features and a Logistic regression classifier is replaced by a Linear SVM. Here, we experimented with the regularization parameter C, for which values in range $\{1, 20, 50, 100, 200\}$ were tested. This SVM variant is from this point on referred to as "SVM (Martinc et al.)". As this feature construction pipeline consists of too many parameters, we were not able to perform extensive grid search due to computational complexity. Thus, we did not experiment with feature construction parameters, and kept the configuration proposed in the original study.

Linear SVM with automatic feature construction. The second learner is a libSVM linear classifier [65], trained on a predefined number of word and character level n-grams, constructed using Scikit-learn's *TfidfVectorizer* method. To find the best setting, we varied the SVM's C parameter in range $\{1, 20, 50, 100, 200\}$, the number of word features between $\{10000, 50000, 100000, 200000\}$ and character features between $\{0, 30\}$[8]. Note that the word features were sorted by decreasing frequency. Here, we considered (word) n-grams of lengths between two and six. This SVM variation is from this point on referred to as "SVM (generic)". The main difference between "SVM (generic)" and "SVM (Martinc et al.)" is that the latter approach also considers punctuation-based and suffix-based features. Further, it is capable of constructing features that represent document sentiment, which was proven to work well for social media data

[8] In Figure C.9 (Appendix C), the reader can observe the results of the initial experiments on the number of word features that led to selection of this hyperparameter range.

sets (e.g., tweets). Finally, Martinc's approach also accounts for character repetitions and has a parameter for social-media text cleaning in preprocessing. Note that for both SVM approaches we fine-tuned the hyperparameter C, as is common when employing SVMs. The hyperparameter's values govern how penalized the learner is for a miss-classified instance, which is a property that was shown to vary across data sets (see for example [66]).

Hierarchical attention networks (HILSTM). The first neural network baseline is the recently introduced hierarchical attention network [67]. Here, we performed a grid search over $\{64, 128, 256\}$ hidden layers sizes, embedding sizes of $\{128, 256, 512\}$, batch sizes of $\{8, 24, 52\}$ and number of epochs $\{5, 15, 20, 30\}$. For detailed explanation of the architecture, please refer to the original contribution [67]. We discuss the best-performing architecture in Section 5 below.

Deep feedforward neural networks. As tax2vec constructs feature vectors, we also attempted to use them as inputs for a standard feedforward neural network architecture [68, 69]. Here, we performed a grid search across hidden layer settings: $\{(128, 64), (10, 10, 10)\}$ (where for example $(128, 64)$ corresponds to a two hidden layer neural network, where in the first hidden layer there are 128 neurons and 64 in the second), batch sizes $\{8, 24, 52\}$ and the number of training epochs $\{5, 15, 20\}$.[9]

4.3. Semantic features

In addition to the semantic features constructed by tax2vec, doc2vec-based semantic features [71] were used as a baseline in order to allow for a simple comparison between two semantic feature construction approaches. They were concatenated with the features constructed by Martinc et al.'s SVM approach

[9]The two deep architectures were implemented using TensorFlow [70], and trained using a Nvidia Tesla K40 GPU. We report the best result for top 30 semantic features with the rarest terms heuristic.

described in Section 4.2, in order to compare the benefits merging the BoW-based representations with a different type of semantic features (embedding-based ones). We set the embedding dimension to 256, as it was shown that lower dimensional embeddings do not perform well [72].

4.4. Description of the experiments

The experiments were set up as follows. For the drug-related data sets, we used the splits given in the original paper [21]. For other data sets, we trained the classifiers using stratified 90% : 10% splits. For each classifier, 10 such splits were obtained. The measure used in all cases is F_1, where for the multiclass problems (e.g., MBTI), we use the micro-averaged F_1. All experiments were repeated five times using different random seeds. The features obtained using tax2vec are used in combination with SVM classifiers, while the other classifiers are used as baselines.[10]

5. Classification results

In this section we provide the results obtained by conducting the experiments outlined in the previous section. We begin by discussing the overall classification performance with respect to different heuristics used. Next, we discuss how tax2vec augments the learner's ability to classify when the number of text segments per user is reduced.

5.1. Classification performance evaluation

The F_1 results are presented in Table 2. The first observation is that combining BoW-based representations with semantic features (tax2vec or doc2vec) leads to performance improvements in five out of six cases (MBTI being the only data set where no improvement is detected). Tax2vec outperforms doc2vec-based vectors in three out of five data sets (PAN 2016 (Age), BBC News and

[10]Note that simple feedforward neural networks could also be used in combination with hypernym features—we leave such computationally expensive experiments for further work.

Drugs (effect)), while doc2vec-based features outperform tax2vec on two data sets (PAN 2017 (gender) and Drugs (Side)).

When it comes to tax2vec, up to 100 semantic features aid the SVM learners to achieve better accuracy. The most apparent improvement can be observed for the case of PAN 2016 (Age) data set, where the task was to predict age. Here, 10 semantic features notably improved the classifiers' performance (up to approximately 7% for SVM (generic)). Further, a minor improvement over the state-of-the-art was also observed on the PAN 2017 (Gender) data set and the BBC news categorization (see results for SVM (Martinc et al.)). Hierarchical attention networks outperformed all other learners for the task of side effects prediction, yet semantics-augmented SVMs outperformed neural models when general drug effects were considered as target classes. Similarly, no performance improvements were offered by tax2vec on the MBTI data set.

Table 2: Effect of the added semantic features to classification performance, where all text segments (tweets/comments per user or segments per news article) are used. The best performing feature selection heuristic for the majority of top performing classifiers was "rarest terms" or "Closeness centrality", indicating that only a handful of hypernyms carry added value, relevant for classification. Note that the results in the table correspond to the best performing combination of a classifier and a given heuristic.

# Semantic	Learner	PAN (Age)	PAN (Gender)	MBTI	BBC News	Drugs (effect)	Drugs (side)
0	HLSTM	0.422	0.752	0.407	0.833	0.443	0.514
0	SVM (Martinc et al.)	0.417	0.814	0.682	0.983	0.468	0.503
0	SVM (generic)	0.424	0.751	0.556	0.967	0.445	0.462
256 (doc2vec)	SVM (Martinc et al.)	0.422	0.817	0.675	0.979	0.416	0.523
30 (tax2vec)	DNN	0.400	0.511	0.182	0.353	0.400	0.321
10 (tax2vec)	SVM (Martinc et al.)	0.445	0.815	0.679	0.996	0.47	0.506
	SVM (generic)	0.502	0.781	0.556	0.972	0.445	0.469
25 (tax2vec)	SVM (Martinc et al.)	0.454	0.814	0.681	0.984	0.468	0.500
	SVM (generic)	0.484	0.755	0.554	0.967	0.449	0.466
50 (tax2vec)	SVM (Martinc et al.)	0.439	0.814	0.681	0.983	0.462	0.499
	SVM (generic)	0.444	0.751	0.554	0.963	0.446	0.463
100 (tax2vec)	SVM (Martinc et al.)	0.424	0.816	0.678	0.984	0.466	0.496
	SVM (generic)	0.422	0.749	0.551	0.958	0.443	0.46
500 (tax2vec)	SVM (Martinc et al.)	0.383	0.797	0.662	0.975	0.45	0.477
	SVM (generic)	0.400	0.724	0.532	0.909	0.424	0.438
1000 (tax2vec)	SVM (Martinc et al.)	0.368	0.783	0.647	0.964	0.436	0.466
	SVM (generic)	0.373	0.701	0.512	0.851	0.407	0.420

We now present the classification results in the form of critical distance diagrams, shown in Figures 4, 5 and 6. The diagrams show average ranks of different algorithms according to the (micro) F_1 measure. A red line connects groups of classifiers that are not statistically significantly different from each other at a confidence level of 5%. The significance levels are computed using Friedman multiple test comparisons followed by Nemenyi post-hoc correction [73]. For each data set, we selected the best performing parametrization (hyperparameter settings). The best (on average) performing C parameter for both SVM models was 50. The number of features that performed the best for all hyperparameter settings of the SVM (generic) considered in this study is 100,000. The HILSTM architecture's topology varied between data sets, yet we observed that the best results were obtained when more than 15 epochs of training were conducted, combined with the hidden layer size of 64 neurons, where the size of the attention layer was of the same dimension.

In terms of feature selection, the following can be observed (Figure 4). On average, the best performance was obtained when rarest terms heuristic was considered (first and fifth rank). Further, rarest terms, as well as the Personalized PageRank performed better (on average) than mutual information, which can be considered as a baseline in this comparison. The results indicate that my-

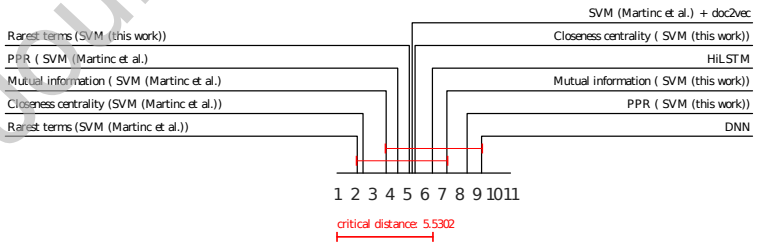


Figure 4: Average overall classifier ranks. The top (on average) performing classifier is an SVM (Martinc et al.) classifier augmented with semantic features, selected using either simple frequency counts or closeness centrality.

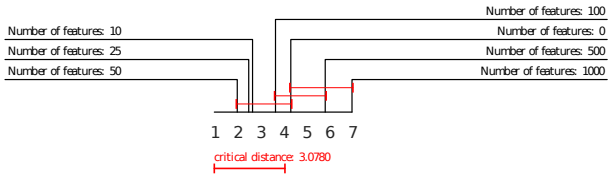


Figure 5: Effect of semantic features on average classifier rank. Up to 100 semantic features positively affects the classifiers' performance.

Figure 6: Overall model performance. SVMs dominate the short text classification. The diagram shows performance averaged over all data sets, where the best model parameterizations (see Table 2) were used for comparison.

opic feature selection is not optimal when considering novel semantic features. We can also observe that on average the configuration with doc2vec semantic features (SVM (Martinc et al.) + doc2vec) performs worse (ranking as sixth) than all other configurations with SVM (Martinc et al.).

In Figure 5, the reader can observe the performances *of all learners*, averaged w.r.t. to the number of semantic features. The drawn diagram indicates that adding 10, 25 or 50 features to a classifier perform similarly well, however, as also discussed in the previous paragraph, the performance drops when larger semantic space is considered.

Finally, in Figure 6 it can be observed that the overall performance of Martinc et al.'s SVMs is the best, followed by generic SVMs, as well as HILSTMs. We believe such performance drop with deep neural networks in general is due to concatenation of documents prior to learning, and as only a fixed sequence

Table 3: Effect of added semantic features to classification performance—few shot learning.

Semantic (tax2vec)	Learner	PAN (Age)	PAN (Gender)	MBTI	BBC News	Drugs (effect)	Drugs (side)
0	SVM (Martinc et al.)	0.378	0.617	0.288	0.977	0.468	0.503
	SVM (generic)	0.429	0.554	0.225	0.936	0.445	0.462
10	SVM (Martinc et al.)	0.39	0.616	0.292	0.981	0.47	0.503
	SVM (generic)	0.429	0.557	0.225	0.948	0.444	0.464
25	SVM (Martinc et al.)	0.429	0.618	0.288	0.979	0.465	0.5
	SVM (generic)	0.439	0.562	0.226	0.933	0.445	0.458
50	SVM (Martinc et al.)	0.402	0.617	0.288	0.974	0.474	0.504
	SVM (generic)	0.42	0.557	0.225	0.919	0.442	0.46
100	SVM (Martinc et al.)	0.382	0.614	0.286	0.974	0.476	0.493
	SVM (generic)	0.411	0.552	0.223	0.906	0.437	0.457
500	SVM (Martinc et al.)	0.359	0.604	0.276	0.959	0.465	0.471
	SVM (generic)	0.365	0.548	0.22	0.8	0.419	0.435
1000	SVM (Martinc et al.)	0.34	0.59	0.266	0.925	0.442	0.46
	SVM (generic)	0.359	0.535	0.213	0.704	0.412	0.417

length can be considered, potentially large parts of the token space were neglected during learning. A similar result was, for example observed in the most recent PAN competition [74].

5.2. Few-shot (per instance) learning

As discussed in the introductory sections, one of the goals of this paper was also to explore the setting, where only a handful of text segments per user are considered. Even though such setting is not strictly a few-shot learning [25], reducing the number of text segments per instance (e.g., user) aims to simulate a setting where there is limited information available. In Table 3, we present the results for the setting, where only (up to) 10 text segments (e.g., tweets or paragraphs in a given news article) were used for training.

The segments were sampled randomly. Only a single text segment per user was considered for the medical texts, as they consist of at max of three commentaries. Similarly, as the BBC news data set consists of news article-genre pairs, we split the news article to paragraphs, which we randomly sampled. The rationale for such sampling is to be able to evaluate tax2vec's performance when, for example, only a handful of paragraphs are available (e.g., only the lead).

We observe that tax2vec based features improve the learners' performance on all of the data sets, albeit by a small margin. The results indicate that

adding semantic information improves the performance as only a handful of text segments does not necessarily contain the relevant information.

5.3. Few-shot learning results

We next discuss the results of few-shot learning, as to our knowledge this type of experiments were not conducted before in combination with semantic feature construction methods. The first observation is, semantic features indeed offer more consistent performance improvements than those observed in Table 4, where incremental improvements were not observed on all data sets. In a few-shot learning scenario, however, on all data sets, the inclusion of semantic space either resulted in similar or better performance, indicating a consistent positive effect on the learning in a limited setting. The differences in learner's performance vary around 1% improvement. For example, a 1% improvement was observed for PAN 2016 (Age), BBC News and MBTI data sets.

We finally comment on the classification performance when considering the BBC data set when comparing to reported state-of-the-art. The observed results ($\geq$98%) are competitive to neural approaches, such as for example as reported in [75], where similar span of accuracy was observed. Furthermore, doc2vec-based models have been observed to perform similarly [76]. The results of this work indicate that by considering smaller number of paragraphs (instead of whole documents), competitive performance can be observed on the BBC data set.

5.4. Interpretation of results

In this section we explain the intuition behind the effect of semantic features on the classifier's performance. Note that the best performing SVM models consisted of thousands of tf-idf word and character level features, yet only up to 100 semantic features, when added, notably improved the performance. This effect can be understood via the way SVMs learn from high-dimensional data. With each new feature, we increase the dimensionality of the feature space. Even a single feature, when added, potentially impacts the hyperplane construction. Thus, otherwise problem-irrelevant features can become relevant when novel

features are added. We believe that adding semantic features to (raw) word tf-idf vector space introduces new information, crucial for successful learning, and potentially aligns the remainder of features so that the classifier can better separate the points of interest.

The other explanation for the notable differences in predictive performance is possibly related to small data set sizes, where only a handful of features can be of relevance and thus notably impact a given classifier's performance. We next discuss the impact of the number of selected semantic features on performance.

5.5. How large semantic space should be considered?

Tables 3 and 4 show that a relatively small number of semantic features are needed for potential performance gains. Note that the number of semantic features that need to be considered is around ≤ 100 in most of the cases. The results indicate that a relatively small proportion of the semantic space carries relevant (additional) information, whereas the remainder potentially introduces noise that degrades the performance. Note that in the limit every term from the taxonomy derived from a given corpus could be considered. In such a scenario, many terms would be irrelevant and would only introduce noise. The experiments conducted in this paper indicate that the threshold for the number of features is in the order of hundreds, yet not more features.

6. Qualitative assessment and explainability of tax2vec

This section discusses the properties of the resulting semantic space in Section 6.1, which is followed by a discussion on the explainability of the proposed tax2vec algorithm in Section 6.2.

6.1. Analysis of the resulting semantic space

In this section we discuss the qualitative properties of the obtained corpus-based taxonomies. We present the results concerning hypernym frequency distributions, as well as the overall structure of an example corpus-based taxonomy.

As the proposed approach is entirely symbolic—each feature can be traced back to a unique hypernym—we explored the feature space qualitatively by exploring the statistical properties of the induced taxonomy using graph-statistical approaches. Here, we modeled hypernym frequency distributions to investigate possible similarity with the Zipf's law [77]. The analysis was performed using the Py3plex library [78]. We also visualized the document-based taxonomy of the PAN 2016 (Age) data set using Cytoscape [79].

The examples in this section are all based on the corpus-based taxonomy, constructed from the PAN 2016 (Age) data set. The results of fitting various heavy-tailed distributions to the hypernym frequencies are given in Figure 7.

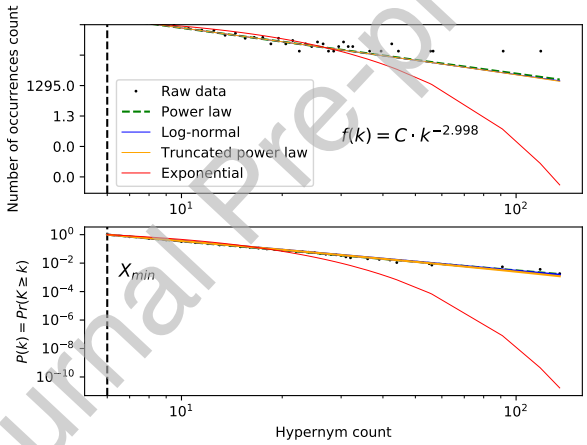


Figure 7: Hypernym frequency distribution for the PAN 2016 (Age) data set. The equation above the upper plot denotes the coefficients of a power law distribution (C is a constant). In real world phenomena, the exponent of the rightmost expression was observed to range between ≈ 2 and ≈ 3, indicating the hypernym structure of the feature space is subject to a heavy-tailed (possibly best fit—power law) distribution. The X_{min} denotes the hypernym count, after which notable differences in hypernym counts—scale free behavior is observed. Such distribution is to some extent expected, as some hypernyms are more general than others, and thus present in more document-hypernym mappings.

We fitted power law, truncated normal, log-normal and exponential distribu-

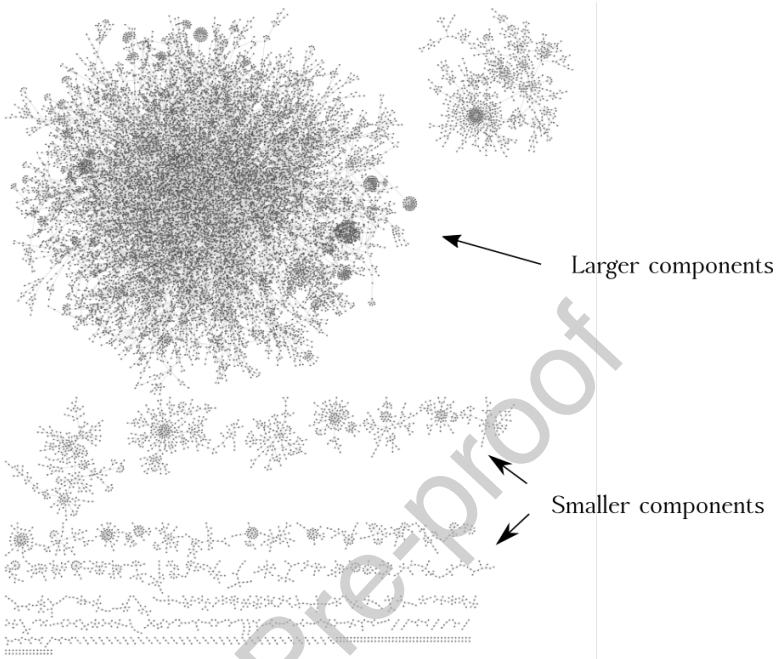


Figure 8: Topological structure of the hypernym space, induced from the PAN 2016 (Age) data set. Multiple connected components emerged, indicating not all hypernyms map to the same high-level concepts. Such segmentation is data set-specific, and can also potentially provide the means to compare semantic spaces of different data sets. It can be observed that the obtained space is organized in multiple separate components. The largest are drawn at the topmost part of the figure, whereas the smaller ones at the bottom. Such segmentation corresponds to generalizations based on different parts of speech, e.g., nouns and verbs.

tions to the hypernym frequency data. For detailed overview of the distributions we refer the reader to [80]. One of the key properties we researched was whether the underlying hypernym distribution is exponential or not, as non-exponential distributions indicate similarity with the well known Zipf's law [77]. The hypernym corpus-based taxonomy is visualized in Figure 8.

Here, each node represents a hypernym obtained in word-to-hypernym mapping phase of tax2vec. The edges represent the hypernymy relation between a given pair of hypernyms.

We next present the results of modeling the corpus-based hypernym frequency distributions. The two functions representing the best fit to hypernym frequency distributions are indeed the power law and the truncated power law. As similar behavior is observed for word frequency in documents [77], we believe hypernym distributions are a natural extension, as naturally, if a high-frequency word maps to a given hypernym, the hypernym will be relatively more common with respect to the occurrence of other hypernyms.

We observe that multiple connected components of varying sizes emerge. There exists only a single largest connected component, which consists of more general noun hypernyms, such as *entity* and similar. Interestingly, many smaller components also emerged, indicating parts of the word vector space could be mapped to very specific, disconnected parts of the WordNet taxonomy. Some examples of small disconnected components include (one component per line), indicating also verb-level semantics can be captured and taken into account:

$'spot.v.02', 'discriminate.v.03'' homestead.v.01', 'settle.v.21'$
$'smell.v.05', 'perceive.v.02', 'understand.v.02'$
$'dazzle.v.01', 'blind.v.01'$
$'romance.v.02', 'adore.v.01', 'care\_for.v.02', 'love.v.03', 'love.v.01'$
$'surrender.v.01', 'yield.v.12', 'capitulate.v.01'$

6.2. Explainability of tex2vec

As discussed in the previous sections, tax2vec selects a set of hypernyms according to a given heuristic and uses them for learning. One of the key benefits of such approach is that the selected semantic features can easily be inspected, hence potentially offering interesting insights into the semantics, underlying the problem at hand.

We discuss here a set of 30 features which emerged as relevant according to the "mutual information" heuristic when the BBC News and PAN 2016 (Age) data sets were considered. Here, tax2vec was trained on 90% of the data, the rest was removed (test set). The features and their corresponding mutual information scores are shown in Table 4.

Table 4: Most informative features with respect to the target class (ranked by MI)—Classes represent news topics (BBC) and different age intervals (PAN 2016 (Age)). Individual target classes are sorted according to a descending mutual information with respect to a given feature.

		Sorted target class-mutual information pairs				
Semantic feature	Average MI	Class 1	Class 2	Class 3	Class 4	Class 5
BBC News data set						
tory.n.03	0.057	politics:0.14	entertainment:0.05	business:0.03	sport:0.01	x
movie.n.01	0.059	business:0.14	politics:0.04	entertainment:0.04	sport:0.02	x
conservative.n.01	0.061	politics:0.15	entertainment:0.05	business:0.03	sport:0.01	x
vote.n.02	0.061	business:0.15	entertainment:0.04	politics:0.04	sport:0.02	x
election.n.01	0.063	entertainment:0.16	business:0.05	politics:0.04	sport:0.0	x
topology.n.04	0.063	entertainment:0.16	business:0.05	politics:0.04	sport:0.0	x
mercantile_establishment.n.01	0.068	politics:0.17	business:0.07	entertainment:0.03	sport:0.01	x
star_topology.n.01	0.069	politics:0.17	business:0.07	entertainment:0.03	sport:0.01	x
rightist.n.01	0.074	politics:0.18	business:0.06	entertainment:0.04	sport:0.01	x
marketplace.n.02	0.087	entertainment:0.22	business:0.06	politics:0.05	sport:0.01	x
PAN (Age) data set						
hippie.n.01	0.007	25-34:0.01	35-49:0.01	18-24:0.0	65-xx:0.0	50-64:0.0
ceremony.n.03	0.007	25-34:0.01	35-49:0.01	18-24:0.01	65-xx:0.0	50-64:0.0
resource.n.02	0.008	50-64:0.02	18-24:0.01	25-34:0.0	65-xx:0.0	35-49:0.0
draw.v.07	0.008	25-34:0.02	35-49:0.01	50-64:0.01	65-xx:0.0	18-24:0.0
observation.n.02	0.008	25-34:0.02	35-49:0.01	50-64:0.01	65-xx:0.0	18-24:0.0
wine.n.01	0.008	35-49:0.02	25-34:0.01	18-24:0.01	50-64:0.01	65-xx:0.0
suck.v.02	0.008	25-34:0.02	50-64:0.02	35-49:0.0	65-xx:0.0	18-24:0.0
sleep.n.03	0.008	25-34:0.02	50-64:0.02	35-49:0.0	65-xx:0.0	18-24:0.0
recognize.v.09	0.009	25-34:0.02	35-49:0.02	18-24:0.0	50-64:0.0	65-xx:0.0
weather.v.04	0.009	25-34:0.02	50-64:0.02	35-49:0.0	18-24:0.0	65-xx:0.0
invention.n.02	0.009	25-34:0.02	35-49:0.01	18-24:0.01	50-64:0.0	65-xx:0.0
yankee.n.03	0.01	50-64:0.02	18-24:0.01	25-34:0.01	35-49:0.0	65-xx:0.0

We can observe that the "sport" topic (BBC data set) is not well associated with the prioritized features. On the contrary, terms such as "rightist" and "conservative" emerged as relevant for classifying into the "politics" class. Similarly, "marketplace" for example, appeared relevant for classifying into the "entertainment" class. Even more interesting associations emerged when the same feature ranking was conducted on the PAN 2016 (Age) data set. Here, terms such as "resource" and "wine" were relevant for classifying middle-aged ("wine") and older adult ("resource") populations. Note that the older population (65-xx class) was not associated with any of the hypernyms. We believe the reason for this is that the number of available tweets decreases with age.

We repeated a similar experiment (BBC data set) using the "rarest terms"

heuristic. The terms which emerged are:

'problem.n.02', 'question.n.02', 'riddle.n.01', 'salmon.n.04', 'militia.n.02', 'orphan.n.04', 'taboo.n.01', 'desertion.n.01', 'dearth.n.02', 'outfitter.n.02', 'scarcity.n.01', 'vasodilator.n.01', 'dilator.n.02', 'fluoxetine.n.01', 'high blood pressure.n.01', 'amlodipine besylate.n.01', 'drain.n.01', 'imperative mood.n.01', 'fluorescent.n.01', 'veneer.n.01', 'autograph.n.01', 'oak.n.02', 'layout.n.01', 'wall.n.01', 'firewall.n.03', 'workload.n.01', 'manuscript.n.02', 'cake.n.01', 'partition.n.01', 'plasterboard.n.01'

Even if the feature selection method is unsupervised (not directly associated to classes), we can immediately observe that the features correspond to different topics, raging from medicine (e.g., high blood presure), politics (e.g., militia), food(e.g., cake) and more, indicating that the rarest hypernyms are indeed diverse and as such potentially useful for the learner.

The results suggest that tax2vec could potentially also be used to inspect the semantic background of a given data set directly, regardless of the learning task. We believe there are many potential uses for the obtained features, including the following, to be addressed in further work.

- Concept drift detection, i.e. topics change over time; could it be qualitatively detected?
- Topic domination, i.e. what type of topic is dominant with respect to e.g., a geographical region inspected?
- What other learning tasks can benefit by using second level semantics? Can the obtained features be used, for example, for fast keyword search?

7. Implementation and availability

The tax2vec algorithm is implemented in Python 3, where Multiprocessing[11], SciPy [81] and Numpy [82] libraries are used for fast (sparse), vectorized

[11] https://docs.python.org/2/library/multiprocessing.html

operations and parallelism.

As performing a grid search over several parameters is computationally expensive, the majority of the experiments were conducted using the SLING supercomputing architecture.[12]

We developed a stand-alone library that relatively seamlessly fits into existing text mining workflows, hence the Scikit-learn's model syntax was adopted [83]. The algorithm is first initiated as an object:

```
vectorizer = tax2vec(heuristic,number of features)
```

followed by standard *fit* and *transform* calls:

```
new_features = vectorizer.fit_transform(corpus, optional labels)
```

Such implementation offers fast prototyping capabilities, needed ubiquitously in the development of learning algorithms and executable NLP and text mining workflows.

The proposed tax2vec approach is freely available as a Python 3 library at `https://github.com/SkBlaz/tax2vec`, which includes also the installation instructions.

8. Conclusions and future work

In this work we propose tax2vec, a parallel algorithm for taxonomy-based enrichment of text documents. Tax2vec first maps the words from individual documents to their hypernym counterparts, which are considered as candidate features and weighted according to a normalized tf-idf metric. To select only a user-specified number of relevant features, tax2vec implements multiple feature selection heuristics, which select only the potentially relevant features. The sparse matrix of constructed features is finally used alongside the bag-of-words document representations for the task of text classification, where we study its

[12] `http://www.sling.si/sling/`

performance on small data sets, where both the number of text segments per user, as well as the number of overall users considered are small.

The tax2vec approach considerably improves the classification performance especially on data sets consisting of tweets, but also on the news. The proposed implementation offers a simple-to-use API, which facilitates inclusion into existing text preprocessing workflows.

As the next step, the tax2vec will be tested on SMS spam data [84], which is another potentially interesting short text data set where taxonomy-based features could improve performance and help the user better understand what classifies as spam (and what not).

One of the drawbacks we plan to address is the support for arbitrary directed acyclic multigraphs—structures commonly used to represent background knowledge. Support for such knowledge would offer a multitude of applications in e.g., biology, where gene ontology and other resources which annotate entities of interest are freely available.

In this work we focus on BoW representation of documents, yet we believe tax2vec could also be used along Continuous Bag-of-Words (CBoW) models. We leave such experimentation for further work.

Even though we use Lesk for the disambiguation task, we believe recent advancements in neural disambiguation [85] could also be a "drop-in" replacement for this part of tax2vec. We leave the exploration of such options for further work.

In this work we explored how WordNet could be adapted for scalable feature construction, however tax2vec is by no means limited to manually curated relational (hierarchical) structures. As part of the further work, we believe feature construction based on *knowledge graphs* could also be an option.

The abundance of neural embedding methods introduced in the recent years can be complementary to tax2vec. Understanding how the performance can be improved by jointly using both tax2vec's features and neural network-based ones is a potential interesting research opportunity. Further, in NLP setting, not much attention is devoted to this topic, thus we believe these results offer

new trajectories for few-shot learning research.

Other further work considers joining the tax2vec features with existing state-of-the-art deep learning approaches, such as the hierarchical attention networks, which are—according to this study—not very suitable for learning on scarce data sets. We believe that the introduction of semantics into deep learning could be beneficial for both performance, as well as the interpretability of currently poorly understood black-box models.

Finally, as the main benefit of tax2vec is its explanatory power, we believe it could be used for fast keyword search; here, for example, new news or articles could be used as inputs, where the ranked list of semantic features could be directly used as candidate keywords.

Acknowledgements

We would first like to thank the reviewers for insightful comments that improved this paper. The work of the first author was funded by the Slovenian Research Agency through a young researcher grant (TSP). The work of other authors was supported by the Slovenian Research Agency (ARRS) core research programme *Knowledge Technologies* (P2-0103), an ARRS funded research project *Semantic Data Mining for Linked Open Data* (financed under the ERC Complementary Scheme, N2-0078) and European Unionś Horizon 2020 research and innovation programme under grant agreement No 825153, project EMBEDDIA (Cross-Lingual Embeddings for Less-Represented Languages in European News Media). This research has received funding from the European Unions Horizon 2020 research and innovation programme under grant agreement No 769661, SAAM project. We also gratefully acknowledge the support of NVIDIA Corporation for the donation of Titan-XP GPU.

References

[1] F. Sebastiani, Machine learning in automated text categorization, ACM Comput. Surv. 34 (1) (2002) 1–47.

[2] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805.

[3] D. Tang, B. Qin, T. Liu, Document modeling with gated recurrent neural network for sentiment classification, in: Proceedings of the 2015 conference on empirical methods in natural language processing, 2015, pp. 1422–1432.

[4] M. Kusner, Y. Sun, N. Kolkin, K. Weinberger, From word embeddings to document distances, in: International Conference on Machine Learning, 2015, pp. 957–966.

[5] F. Rangel, P. Rosso, M. Potthast, B. Stein, Overview of the 5th author profiling task at pan 2017: Gender and language variety identification in twitter, Working Notes Papers of the CLEF.

[6] A. Ławrynowicz, Semantic Data Mining: An Ontology-based Approach, Vol. 29, IOS Press, 2017.

[7] E. Angelino, N. Larus-Stone, D. Alabi, M. Seltzer, C. Rudin, Learning certifiably optimal rule lists, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 35–44.

[8] A. Vavpetič, N. Lavrač, Semantic subgroup discovery systems and workflows in the sdm-toolkit, The Computer Journal 56 (3) (2013) 304–320.

[9] M. Perovšek, A. Vavpetič, B. Cestnik, N. Lavrač, A wordification approach to relational data mining, in: J. Fürnkranz, E. Hüllermeier, T. Higuchi (Eds.), Discovery Science, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 141–154.

[10] P. R. Adhikari, A. Vavpetič, J. Kralj, N. Lavrač, J. Hollmén, Explaining mixture models through semantic pattern mining and banded matrix visualization, Machine Learning 105 (1) (2016) 3–39.

[11] S. Scott, S. Matwin, Text classification using wordnet hypernyms, Usage of WordNet in Natural Language Processing Systems.

[12] C. Kim, P. Yin, C. X. Soto, I. K. Blaby, S. Yoo, Multimodal biological analysis using NLP and expression profile, in: 2018 New York Scientific Data Summit (NYSDS), 2018, pp. 1–4.

[13] S. Chang, W. Han, J. Tang, G.-J. Qi, C. C. Aggarwal, T. S. Huang, Heterogeneous network embedding via deep architectures, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '15, ACM, New York, NY, USA, 2015, pp. 119–128.

[14] L. C. Freeman, Research Methods in Social Network Analysis, Routledge, 2017.

[15] J. Wang, Z. Wang, D. Zhang, J. Yan, Combining knowledge with deep convolutional neural networks for short text classification, in: Proceedings of IJCAI, Vol. 350, 2017, p. online.

[16] M. Chen, X. Jin, D. Shen, Short text classification improved by learning multi-granularity topics, in: Twenty-Second International Joint Conference on Artificial Intelligence, 2011.

[17] F. Rangel, P. Rosso, I. Chugur, M. Potthast, M. Trenkmann, B. Stein, B. Verhoeven, W. Daelemans, Overview of the 2nd author profiling task at PAN 2014, in: Working Notes Papers of the CLEF 2014, 2014, pp. 1–30.

[18] F. Rangel, P. Rosso, B. Verhoeven, W. Daelemans, M. Potthast, B. Stein, Overview of the 4th author profiling task at pan 2016: cross-genre evaluations, in: Working Notes Papers of the CLEF 2016 Evaluation Labs. CEUR Workshop Proceedings/Balog, Krisztian [edit.]; et al., 2016, pp. 750–784.

[19] Z. Chu, S. Gianvecchio, H. Wang, S. Jajodia, Detecting automation of twitter accounts: Are you a human, bot, or cyborg?, IEEE Transactions on Dependable and Secure Computing 9 (6) (2012) 811–824.

[20] Z. Chu, S. Gianvecchio, H. Wang, S. Jajodia, Who is tweeting on twitter: human, bot, or cyborg?, in: Proceedings of the 26th annual computer security applications conference, ACM, 2010, pp. 21–30.

[21] F. Grässer, S. Kallumadi, H. Malberg, S. Zaunseder, Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning, in: Proceedings of the 2018 International Conference on Digital Health, DH '18, ACM, New York, NY, USA, 2018, pp. 121–125.

[22] R. Boyce, G. Gardner, H. Harkema, Using natural language processing to identify pharmacokinetic drug-drug interactions described in drug package inserts, in: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing, Association for Computational Linguistics, 2012, pp. 206–213.

[23] J. Cho, K. Lee, E. Shin, G. Choy, S. Do, How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?, arXiv preprint arXiv:1511.06348.

[24] R. Socher, M. Ganjoo, C. D. Manning, A. Ng, Zero-shot learning through cross-modal transfer, in: Proceedings of the Advances in neural information processing systems, 2013, pp. 935–943.

[25] J. Snell, K. Swersky, R. Zemel, Prototypical networks for few-shot learning, in: Advances in Neural Information Processing Systems, 2017, pp. 4077–4087.

[26] T. K. Landauer, Latent Semantic Analysis, Wiley Online Library, 2006.

[27] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, Journal of machine Learning research 3 (1) (2003) 993–1022.

[28] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems 26, Curran Associates, Inc., 2013, pp. 3111–3119.

[29] L. Cagliero, P. Garza, Improving classification models with taxonomy information, Data & Knowledge Engineering 86 (2013) 85–101.

[30] N. Xu, J. Wang, G. Qi, T. S. Huang, W. Lin, Ontological random forests for image classification, in: Computer Vision: Concepts, Methodologies, Tools, and Applications, IGI Global, 2018, pp. 784–799.

[31] M. K. Elhadad, K. M. Badran, G. I. Salama, A novel approach for ontology-based feature vector generation for web text document classification, International Journal of Software Innovation (IJSI) 6 (1) (2018) 1–10.

[32] R. Kaur, M. Kumar, Domain ontology graph approach using markov clustering algorithm for text classification, in: International Conference on Intelligent Computing and Applications, Springer, 2018, pp. 515–531.

[33] T. N. Mansuy, R. J. Hilderman, Evaluating wordnet features in text classification models., in: FLAIRS Conference, 2006, pp. 568–573.

[34] M. Bunge, Causality and Modern Science, Routledge, 2017.

[35] J. Pearl, Causality, Cambridge university press, 2009.

[36] U. Stańczyk, L. C. Jain, Feature selection for Data and Pattern Recognition, Springer, 2015.

[37] A. G. Kakisim, I. Sogukpinar, Unsupervised binary feature construction method for networked data, Expert Systems with Applications 121 (2019) 256 – 265.

[38] B. Škrlj, N. Lavrač, J. Kralj, Symbolic graph embedding using frequent pattern mining, in: P. Kralj Novak, T. Šmuc, S. Džeroski (Eds.), Discovery Science, Springer International Publishing, Cham, 2019, pp. 261–275.

[39] N. Tomašev, K. Buza, K. Marussy, P. B. Kis, Hubness-aware classification, Instance Selection and Feature Construction: Survey and Extensions to Time-series, in: Feature selection for data and pattern recognition, Springer, 2015, pp. 231–262.

[40] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Computers & Electrical Engineering 40 (1) (2014) 16–28.

[41] Z. M. Hira, D. F. Gillies, A review of feature selection and feature extraction methods applied on microarray data, Advances in bioinformatics 2015.

[42] A. Grover, J. Leskovec, Node2vec: Scalable feature learning for networks, in: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, ACM, New York, NY, USA, 2016, pp. 855–864.

[43] Y. Dong, N. V. Chawla, A. Swami, Metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'17, ACM, New York, NY, USA, 2017, pp. 135–144.

[44] S. Jaeger, S. Fulle, S. Turk, Mol2vec: Unsupervised machine learning approach with chemical intuition, Journal of Chemical Information and Modeling 58 (1) (2018) 27–35.

[45] L. F. Ribeiro, P. H. Saverese, D. R. Figueiredo, Struc2vec: Learning node representations from structural identity, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'17, ACM, New York, USA, 2017, pp. 385–394.

[46] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: International Conference on Learning Representations (ICLR), 2017, p. online.

[47] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, in: Advances in Neural Information Processing Systems 30, Curran Associates, Inc., 2017, pp. 1024–1034.

[48] F. Železnỳ, N. Lavrač, Propositionalization-based relational subgroup discovery with RSD, Machine Learning 62 (1-2) (2006) 33–63.

[49] M. Perovšek, A. Vavpetič, B. Cestnik, N. Lavrač, A wordification approach to relational data mining, in: International Conference on Discovery Science, Springer, 2013, pp. 141–154.

[50] F. Rangel, P. Rosso, L. Cappellato, N. Ferro, H. Müller, D. Losada, Overview of the 7th author profiling task at pan 2019: Bots and gender profiling, in: CLEF, 2019, p. online.

[51] G. A. Miller, Wordnet: A lexical database for english, Commun. ACM 38 (11) (1995) 39–41.

[52] A. Gonzalez-Agirre, E. Laparra, G. Rigau, Multilingual central repository version 3.0: upgrading a very large lexical knowledge base, in: Proceedings of the 6th Global WordNet Conference (GWC 2012), Matsue, 2012, p. online.

[53] R. Navigli, Word sense disambiguation: A survey, ACM Comput. Surv. 41 (2) (2009) 10:1–10:69.

[54] P. Basile, A. Caputo, G. Semeraro, An enhanced lesk word sense disambiguation algorithm through a distributional semantic model, in: Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, 2014, pp. 1591–1600.

[55] C. D. Manning, P. Raghavan, H. Schtze, Scoring, term weighting, and the vector space model, Cambridge University Press, 2008, Ch. first, p. 100123.

[56] U. Brandes, A faster algorithm for betweenness centrality, The Journal of Mathematical Sociology 25 (2) (2001) 163–177.

[57] H. Peng, F. Long, C. Ding, Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Transactions on pattern analysis and machine intelligence 27 (8) (2005) 1226–1238.

[58] J. Kralj, M. Robnik-Sikonja, N. Lavrac, NetSDM: Semantic data mining with network analysis, Journal of Machine Learning Research 20 (32) (2019) 1–50.

[59] J. Kralj, Heterogeneous information network analysis for semantic data mining: Doctoral dissertation, Ph.D. thesis, J. Kralj (2017).

[60] Matej Martinc and Iza Škrjanec and Katja Zupan and Senja Pollak, Pan 2017: Author profiling - gender and language variety prediction, in: CLEF, 2017, p. online.

[61] I. B. Myers, The myers-briggs type indicator: Manual (1962).

[62] D. Greene, P. Cunningham, Practical solutions to the problem of diagonal dominance in kernel document clustering, in: Proceedings of the 23rd International Conference on Machine learning (ICML'06), ACM Press, 2006, pp. 377–384.

[63] U. Sapkota, S. Bethard, M. Montes-y-Gómez, T. Solorio, Not all character n-grams are created equal: A study in authorship attribution, in: NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015, 2015, pp. 93–102.

[64] P. K. Novak, J. Smailović, B. Sluban, I. Mozetič, Sentiment of emojis, PloS one 10 (12) (2015) e0144296.

[65] C.-C. Chang, C.-J. Lin, Libsvm: a library for support vector machines, ACM transactions on intelligent systems and technology (TIST) 2 (3) (2011) 27.

[66] D. Meyer, F. Leisch, K. Hornik, The support vector machine under test, Neurocomputing 55 (1) (2003) 169 – 186, support Vector Machines. doi:https://doi.org/10.1016/S0925-2312(03)00431-4.

[67] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, Hierarchical attention networks for document classification, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 1480–1489.

[68] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (7553) (2015) 436.

[69] J. Schmidhuber, Deep learning in neural networks: An overview, Neural networks 61 (2015) 85–117.

[70] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow.org (2015).

[71] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: International conference on machine learning, 2014, pp. 1188–1196.

[72] J. Pennington, R. Socher, C. Manning, Glove: Global vectors for word representation, in: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), 2014, pp. 1532–1543.

[73] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine learning research 7 (Jan) (2006) 1–30.

[74] M. Martinc, B. Škrlj, S. Pollak, Fake or not: Distinguishing between bots, males and females, in: Proceedings of the Tenth International Conference of the CLEF Association (CLEF 2019), 2019, p. online.

[75] M. N. Asim, M. U. G. Khan, M. I. Malik, A. Dengel, S. Ahmed, A robust hybrid approach for textual document classification, arXiv preprint arXiv:1909.05478.

[76] L. Q. Trieu, H. Q. Tran, M.-T. Tran, News classification from social media using twitter-based doc2vec model and automatic query expansion, in: Proceedings of the Eighth International Symposium on Information and Communication Technology, 2017, pp. 460–467.

[77] S. T. Piantadosi, Zipfs word frequency law in natural language: A critical review and future directions, Psychonomic bulletin & review 21 (5) (2014) 1112–1130.

[78] B. Škrlj, J. Kralj, N. Lavrač, Py3plex: A library for scalable multilayer network analysis and visualization, in: Complex Networks and Their Applications VII, Springer International Publishing, Cham, 2019, pp. 757–768.

[79] P. Shannon, A. Markiel, O. Ozier, N. S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, T. Ideker, Cytoscape: a software environment for integrated models of biomolecular interaction networks, Genome research 13 (11) (2003) 2498–2504.

[80] S. Foss, D. Korshunov, S. Zachary, et al., An introduction to Heavy-tailed and Subexponential Distributions, Vol. 6, Springer, 2011.

[81] E. Jones, T. Oliphant, P. Peterson, SciPy: Open source scientific tools for Python, http://www.scipy.org/ (2001–).

[82] S. v. d. Walt, S. C. Colbert, G. Varoquaux, The numpy array: a structure for efficient numerical computation, Computing in Science & Engineering 13 (2) (2011) 22–30.

[83] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., Scikit-learn: Machine learning in python, Journal of machine learning research 12 (Oct) (2011) 2825–2830.

[84] S. J. Delany, M. Buckley, D. Greene, Sms spam filtering: Methods and data, Expert Systems with Applications 39 (10) (2012) 9899–9908.

[85] I. Iacobacci, M. T. Pilehvar, R. Navigli, Embeddings for word sense disambiguation: An evaluation study, in: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, Vol. 1, 2016, pp. 897–907.

Appendix A. Personalized PageRank algorithm

The Personalized PageRank (PPR) algorithm is described below. Let V represent the nodes of the corpus-based taxonomy. For each node $u \in V$, a feature vector is computed by calculating the stationary distribution of a random walk, starting at node u. The stationary distribution is approximated by using power iteration, where the i-th component of the approximation in the k-th iteration is computed as

$$\gamma_u(i)^{(k+1)} = \alpha \cdot \sum_{j \to i} \frac{\gamma_u(j)^{(k)}}{d_j^{out}} + (1-\alpha) \cdot v_u(i); k = 1, 2, \ldots \quad \text{(A.1)}$$

The number of iterations k is increased until the stationary distribution converges to the stationary distribution vector (PPR value for node i). In the above equation, α is the damping factor that corresponds to the probability that a random walk follows a randomly chosen outgoing edge from the current node rather than restarting its walk. The summation index j runs over all nodes of the network that have an outgoing connection toward j, (denoted as $j \to i$ in the sum), and d_j^{out} is the out degree of node d_j. The term $v_u(i)$ is the restart distribution that corresponds to a vector of probabilities for a walker's return to the starting node u, i.e. $v_u(u) = 1$ and $v_u(i) = 0$ for $i \neq u$. This vector guarantees that the walker will jump back to the starting node u in case of a restart.[13]

[13]Note that if the binary vector were instead composed exclusively of ones, the iteration would compute the global PageRank vector, and Equation A.1 would correspond to the standard PageRank iteration.

Appendix B. Example document split

While for the data sets consisting of tweets and short comments, the number of segments in a document corresponds to the number of tweets or comments by a user, in the news data set, we varied the size of the news (to create short documents) by splitting the news into paragraphs (we denote such paragraph splits with ———). An example of segmentation of a news from the BBC data set[14] is listed below.

——— The decision to keep interest rates on hold at 4.75% earlier this month was passed 8-1 by the Bank of England's rate-setting body, minutes have shown.——— One member of the Bank's Monetary Policy Committee (MPC) - Paul Tucker - voted to raise rates to 5%. The news surprised some analysts who had expected the latest minutes to show another unanimous decision. Worries over growth rates and consumer spending were behind the decision to freeze rates, the minutes showed. The Bank's latest inflation report, released last week, had noted that the main reason inflation might fall was weaker consumer spending.——— However, MPC member Paul Tucker voted for a quarter point rise in interest rates to 5%. He argued that economic growth was picking up, and that the equity, credit and housing markets had been stronger than expected.——— The Bank's minutes said that risks to the inflation forecast were "sufficiently to the downside" to keep rates on hold at its latest meeting. However, the minutes added: "Some members noted that an increase might be warranted in due course if the economy evolved in line with the central projection". Ross Walker, UK economist at Royal Bank of Scotland, said he was surprised that a dissenting vote had been made so soon. He said the minutes appeared to be "trying to get the market to focus on the possibility of a rise in rates". "If the economy pans out as they expect then they are probably going to have to hike rates." However, he added, any rate increase is not likely to happen until later

[14] https://github.com/suraj-deshmukh/BBC-Dataset-News-Classification/blob/master/dataset/dataset.csv

this year, with MPC members likely to look for a more sustainable pick up in consumer spending before acting.

This news article is split by a parser into the following four segments (and in short document setting only one paragraph is used to represent the document).

- The decision to keep interest rates on hold at 4.75% earlier this month was passed 8-1 by the Bank of England's rate-setting body, minutes have shown.
- One member of the Bank's Monetary Policy Committee (MPC) - Paul Tucker - voted to raise rates to 5%. The news surprised some analysts who had expected the latest minutes to show another unanimous decision. Worries over growth rates and consumer spending were behind the decision to freeze rates, the minutes showed. The Bank's latest inflation report, released last week, had noted that the main reason inflation might fall was weaker consumer spending.
- However, MPC member Paul Tucker voted for a quarter point rise in interest rates to 5%. He argued that economic growth was picking up, and that the equity, credit and housing markets had been stronger than expected.
- The Bank's minutes said that risks to the inflation forecast were"sufficiently to the downside" to keep rates on hold at its latest meeting. However, the minutes added: "Some members noted that an increase might be warranted in due course if the economy evolved in line with the central projection." Ross Walker, UK economist at Royal Bank of Scotland, said he was surprised that a dissenting vote had been made so soon. He said the minutes appeared to be "trying to get the market to focus on the possibility of a rise in rates." "If the economy pans out as they expect then they are probably going to have to hike rates." However, he added, "any rate increase is not likely to happen until later this year, with MPC members likely to look for a more sustainable pick up in consumer spending before

acting."

Appendix C. Impact of different number of features across data sets

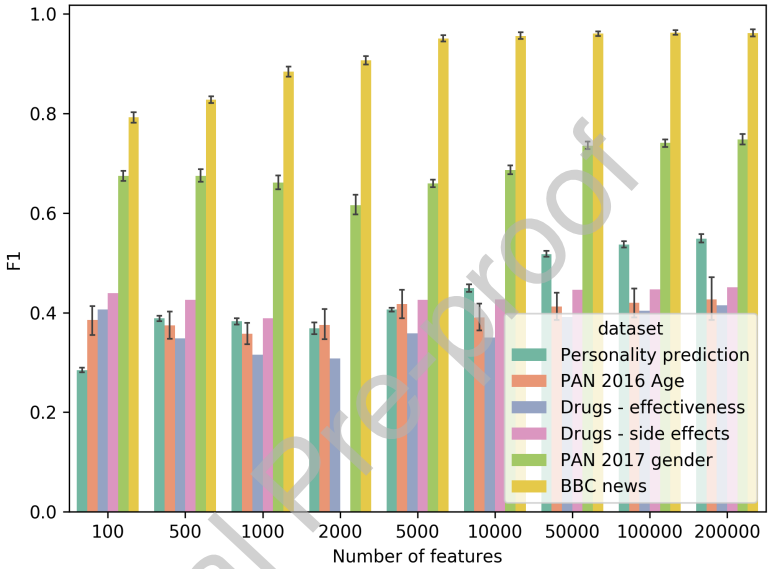


Figure C.9: Impact of the number of features used by the SVM (generic) on the F1 performance. The best performances were observed for feature numbers (word tokens) $\geq$ 10,000, hence these feature numbers were considered in the more expensive experiment stage with semantic vectors.