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

D1.4: Initial deep network architecture (T1.3)

Executive summary
Most of the standard neural network architectures for language processing are adapted to English. As morphologically rich languages might require different inputs and explicit information about morphology, we tested adding such information to the state-of-the-art contextual model BERT. We used a pre-trained multilingual BERT model and combined it with additional morphological information: separately trained universal POS tags, separately trained universal features, as well as prefixes and suffixes. We tested the modified architectures on the named entity recognition problem using three morphologically rich languages: Slovene, Estonian, and Finnish. The tested architectures did not improve the overall results, which indicates that contextual embeddings like BERT (unlike non-contextual embeddings such as fastText) already capture all the necessary information about the morphology. An additional issue addressed in this report is reliability of predictions. Obtaining reliability scores of deep neural network predictions is an important practical issue in text classification. We introduce a novel mechanism using Monte Carlo dropout that successfully addresses the problem. The proposed method is successfully tested on several hate speech datasets using both word and sentence embeddings.

Partner in charge: UL

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:	**D1.4**
Deliverable full title:	**Initial deep network architecture**
Deliverable short title:	**Initial deep networks**
Document identifier:	**EMBEDDIA-D14-InitialDeepNetworks-T13-submitted**
Lead partner short name:	**UL**
Report version:	**submitted**
Report submission date:	**31/12/2019**
Dissemination level:	**PU**
Nature:	**R = Report**
Lead author(s):	**Marko Robnik-Šikonja (UL), Luka Krsnik (UL), Matej Ulčar (UL)**
Co-author(s):	**Matthew Purver (QMUL), Kristiina Vaik (TEXTA), Marko Pranjić (STY)**
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
30/09/2019	v1.0	Marko Robnik-Šikonja (UL)	First draft.
16/10/2019	v1.1	Luka Krsnik (UL)	Added adaptations for morphologically rich languages.
31/10/2019	v1.2	Luka Krsnik (UL)	Added additional results.
05/11/2019	v1.3	Marko Robnik-Šikonja (UL)	Report expanded, checked, and consolidated.
05/11/2019	v1.4	Luka Krsnik (UL)	Added missing data.
20/11/2019	v1.5	Matthew Purver (QMUL)	Updated template, added output section.
27/11/2019	v1.6	Matej Martinc (JSI)	Internal review.
05/12/2019	v1.7	Saturnino Luz (UEDIN	Internal review.
08/12/2019	v1.8	Luka Krsnik, Matej Ulčar, Marko Robnik-Šikonja (UL)	Response to reviews.
13/12/2019	final	Nada Lavrač (JSI)	Report quality checked and finalised.
23/12/2019	submitted	Tina Anžič (JSI)	Report submitted.

Table of Contents

List of abbreviations

NLP	Natural Language Processing
NE	Named Entity
NER	Named Entity Recognition
RNN	Recurrent Neural Network
LSTM network	Long Short-Term Memory network
ELMo	Embeddings from Language Models
BERT	Bidirectional Encoder Representations from Transformers
BNN	Bayesian Neural Networks
MCD	Monte Carlo Dropout

1 Introduction

Deep neural networks are currently the most successful machine learning approach for textual data, beating all other models in practically all language processing and understanding tasks (LeCun et al., 2015; Zhang et al., 2015; Kim et al., 2016; Peters et al., 2018; Devlin et al., 2019). As an input, neural networks require numerical data. Text embeddings provide such an input, ensuring that relations between words are reflected in the distances in numeric space. In this report, we investigate different neural network architectures with the aim to adapt them to specifics of processing morphologically rich languages addressed in the EMBEDDIA project.

The EMBEDDIA project aims to improve the cross-lingual transfer of language resources and trained models using word embeddings and cross-lingual word embeddings. We presented the basic description of embeddings and cross-lingual embeddings in deliverable *D1.1 Datasets, benchmarks and evaluation metrics for cross-lingual word embeddings*, with more details about embeddings and cross-lingual embeddings contained in deliverables *D1.2 Initial cross-lingual and multilingual embeddings technology* and *D1.3 Initial context-dependent and dynamic embeddings technology*. To make this document self-contained, we first repeat some basic explanations in Section 1.1, which a reader acquainted with embeddings can skip. Section 1.2 puts this report in the broader context of project objectives, outlines the specific aims and contributions of this report in the context of task T1.3, and presents the report structure.

1.1 Introducing embeddings

To process text, neural networks require numerical representation of the given text (words, sentences, documents), referred to as **text embeddings**. In this work we focus on **word embeddings**, which are representations of words in numerical form, consisting of vectors of typically several hundred dimensions. The vectors are used as an input to machine learning models; for complex language processing tasks these are typically deep neural networks. The embedding vectors are obtained from specialized learning tasks, based on neural networks, e.g., word2vec (Mikolov, Le, & Sutskever, 2013), GloVe (Pennington et al., 2014), or FastText (Bojanowski et al., 2017). For training, the embedding algorithms use large monolingual text collections (called corpora) to encode important information about word meaning as distances between vectors. In order to enable downstream machine learning on text understanding tasks, the embeddings shall preserve semantic relations between words, and this is true even across languages.

Probably the best known word embeddings are produced by the word2vec method (Mikolov, Sutskever, et al., 2013) which we use as a baseline. The problem with word2vec embeddings is their failure to express polysemous words. During training of an embedding, all senses of a given word (e.g., *paper* as a material, as a newspaper, as a scientific work, and as an exam) contribute relevant information in proportion to their frequency in the training corpus. This causes the final vector to be placed somewhere in the weighted middle of all words' meanings. Consequently, rare meanings of words are poorly expressed with word2vec and the resulting vectors do not offer good semantic representations. For example, none of the 50 closest vectors of the word *paper* is related to science[1].

The idea of **contextual embeddings** is to generate a different vector for each context a word appears in and the context is typically defined sentence-wise. To a large extent, this solves the problems with word polysemy, i.e. the context of a sentence is typically enough to disambiguate different meanings of a word for humans and so it is for the learning algorithms.

In our work, we mostly use, analyze, and improve upon currently the most successful approaches to contextual word embeddings, ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019). Note that the state-of-the-art in embeddings is rapidly progressing (for example, at the time the EMBEDDIA project was conceived, the methods for training contextual embeddings were non-existent). It is therefore possible, that working methods will change during the project duration.

[1] A demo showing near vectors computed with word2vec from Google News corpus is availabe at `http://bionlp-www.utu.fi/wv_demo/`.

Modern word embedding spaces exhibit similar structures across languages, even when considering distant language pairs like English and Vietnamese (Mikolov, Le, & Sutskever, 2013). This means that embeddings independently produced from monolingual text resources can be aligned (Mikolov, Le, & Sutskever, 2013), resulting in a common cross-lingual representation, called the **cross-lingual embedding**, which allows for fast and effective integration of information in different languages.

1.2 Context of the deliverable

The objectives of workpackage WP1 of the EMBEDDIA project are to advance cross-lingual and context-dependent word embeddings and test them with deep neural networks. The specific objectives of T1.3 are to advance deep learning technology for morphologically rich, less-resourced languages Slovene, Croatian, Estonian, Lithuanian, Latvian, Russian, Finnish, and Swedish, as well as for English, i.e. the **nine languages** addressed in EMBEDDIA.

This report describes the results of the work performed in T1.3 from the start of the task in M4 till M12. The main contributions presented in this report (in the order of appearance) are as follows:

1. extensions of LSTM neural networks and current state-of-the-art model BERT with explicit morphology (POS tags, suffixes, prefixes), described in Sections 3.2, 3.3, and 3.4;
2. evaluation of the proposed neural network extensions on a relevant named entity recognition task, described in Section 4;
3. extensions of deep neural architectures for text processing with reliability scores, described in Section 5 and in the appended paper by Miok et al. (2019), published in Proceedings of the International Conference on Statistical Language and Speech Processing 2019.

The above contributions are slightly different from the ones anticipated in the EMBEDDIA project proposal: at the time of proposal writing we namely anticipated that contextual embedding methods will need to be developed from scratch. However, due to recently developed highly successful contextual embedding approaches ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), part of the original EMBEDDIA objectives were already successfully achieved by AllenNLP and Google research groups, which have huge amounts of resources available (AllenNLP developed ELMo and Google developed BERT). Specifically, a variant of BERT, called multilingual BERT, was trained on 104 languages and works well in cross-lingual setting with no need for any further cross-lingual alignments. BERT uses subword input, which is very appropriate for morphologically rich languages (addressing also some of the objectives of Task 1.3).

Given the described developments, we have in task T1.3 partially re-focused our research on the challenging remaining issues in deep neural network architectures. Consequently, we tried to adapt standard LSTM (Long Short-Term Memory) deep neural networks as well as the state-of-the-art contextual neural network architecture BERT for morphologically rich languages. We tried to enrich LSTM networks and BERT with additional information on morphology: separately trained universal POS tags, separately trained universal features, as well as suffixes and prefixes.

This report is split into further four sections. In Section 2 we shortly describe the most successful architectures of deep-neural networks used in natural language processing. In Section 3, we describe extensions to deep neural network architectures we tested in order to improve their performance on morphologically rich languages. We present initial evaluation of the tested extensions in Section 4. Section 5 presents the issue of prediction uncertainty estimation, proposing a novel methodology for reliability assessment of neural classifiers' predictions on textual data, tested on different hate speech datasets. Availability of new resources produced in this work is discussed in Section 6. We present conclusion about the tested extensions to deep neural networks in Section 7 where we also outline plans for further work. Appendix A includes the paper by Miok et al. (2019), published in Proceedings of the International Conference on Statistical Language and Speech Processing 2019.

2 Deep neural networks

Deep learning (LeCun et al., 2015; Goodfellow et al., 2016) differs from other machine learning algorithms by allowing computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. Deep learning methods have improved state-of-the-art results on various tasks, such as visual object recognition, object detection, speech recognition, machine translation, question answering, text classification and many others.

Text is usually treated as a sequence by deep neural networks. Sequences can be of different length and typically there is some dependency between different positions in a sequence (e.g., a verb in a sentence may determine the subsequent choice of nouns). A standard choice of neural network architecture for sequences are recurrent neural networks (RNN) which contain loops. By introducing backward connections, the information from the previous processing steps persists in the network, effectively allowing the network to memorize previous processing, which is well suited for sequences. RNNs are very effective in processing speech, text, signals, and other sequential data. RNNs also introduce many challenges, for example the convergence of learning to stable weights is much slower.

The most popular type of RNNs are Long Short Term Memory (LSTM) networks. LSTM networks allow explicit control over which information is preserved and which one is forgotten. The network learns behavior of its own weights (called cell weights) and the weights of three gates, the input, output, and forget gate. These gates control the flow of information inside the LSTM neuron. The input gate controls the flow of new values into the cell, the forget gate controls the persistence of values in the cell, and the output gate controls which cell values participate in the computation of the output of the LSTM neuron.

Recently, after the conception of the EMBEDDIA project, a new state-of-the-art deep neural network approach to language modeling, contextual embeddings, and classification was introduced. BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) generalises the idea of language models to masked language models—inspired by cloze tests—which test the understanding of a text by removing a certain portion of words that the participant is asked to replace. The masked language model randomly masks some of the tokens from the input, and the task of the language model is to predict the missing token based on its neighbourhood. BERT uses transformer architecture of neural networks (Vaswani et al., 2017), which uses both left and right context in predicting the masked word and further introduces the task of predicting whether two sentences appear in a sequence. The input representation of BERT are sequences of tokens representing subword units. Some very common words are kept as single tokens, others are split into subwords (e.g., common stems, prefixes, suffixes—if needed down to a single letter tokens). The original BERT project offers pre-trained English, Chinese and multilingual models; the latter is trained on 104 languages simultaneously. BERT has shown excellent performance on 11 NLP tasks: 8 from GLUE language understanding benchmark (Wang et al., 2018), question answering, named entity recognition, and common-sense inference.

Rather than training an individual classifier for every classification task, which is resource and time expensive, we use a pre-trained general BERT multilingual language model and fine-tune it on a specific task. Fine-tuning a large pre-trained model to a specific similar similar task is useful and common in modern NLP, because it allows extraction of textual features without development and lengthy training. Frequently, this approach requires less task-specific data. During pre-training, BERT model learns relations between sentences (entailment) and between tokens within a sentence. This knowledge is then used during training on a specific down-stream task (Devlin et al., 2019). The use of BERT in a token classification task only requires adding connections between its last hidden layer and new neurons corresponding to the number of classes in the intended task. To classify a sequence, we usually apply a pooling operation before the classification layer to reduce the sequence length dimension to one. The fine-tuning process is applied to the whole network and all of the parameters of BERT and new class specific weights are fine-tuned jointly to maximize the log-probability of the correct labels.

3 Adaptations of neural networks for morphologically rich languages

In morphologically rich languages, the information about grammatical relations (e.g., subject, predicate, object) is expressed in the morphology of words instead of in particles or relative positions of words. State-of-the-art contextual embeddings like BERT (Devlin et al., 2019) already capture a lot of information contained in morphology and work well for this kind of languages (Pires et al., 2019). As the EMBEDDIA project was conceived before contextual embeddings were discovered, it proposed to directly inject morphological information into deep neural networks. In this report, we test this idea and compare it with modern contextual embeddings, specifically with BERT. The comparison is done on a downstream task of named entity recognition (NER), described in Section 4.1. We use an adapted version of the NER task, which is probably the only realistic language understanding task available for all EMBEDDIA languages. Note that the datasets used and the purpose of the NER task in this report is different from the NER task in Deliverable *D2.2. Initial cross-lingual semantic enrichment technology*. In this report, we are only interested in NER as a benchmark for inclusion of morphological information and not to maximally improve the NE recognition rate (e.g., we do not use any external information, fine-tuning of models etc).

In Section 3.1, we present two baseline models, LSTM with non-contextual embeddings, and BERT contextual model whose last layer can be fine-tuned for a specific task. In Section 3.2, we describe a neural architecture where we inject universal part-of-speech tags into the network. In Section 3.3, we propose an architecture that also uses universal features, and in Section 3.4 we add explicit information on word prefixes and suffixes to a network.

3.1 Baseline models

For evaluation purposes, we built two baseline models and compared them with adapted models. First, we created a neural network that utilises pre-trained non-contextual fastText embeddings (Joulin et al., 2016). The fastText embedding uses subword inputs that are suitable for morphologically rich languages. Besides the embeddings layer that used fastText embeddings of size 300, we utilised one unidirectional LSTM layer with 256 LSTM units, and a final softmax classification layer, as illustrated in Figure 1. We trained the LSTM and classification layer for each specific task using the batch size of 16.

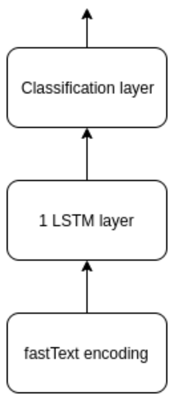

Figure 1: The baseline neural network architecture using fastText embeddings and LSTM cells.

The second baseline model was a pre-trained cased multilingual BERT-Base model with added final softmax classification layer. We fine-tuned the model for our task by back-propagating values through the whole neural network, not just through the classification layer. This is similar to other token classification tasks explained in (Devlin et al., 2019). The structure of the model is shown in Figure 2.

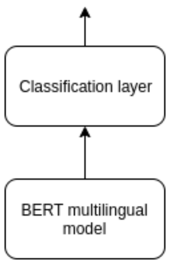

Figure 2: The baseline neural network architecture using BERT for prediction.

For many NLP tasks, BERT-based models are much more successful than the models with classical LSTM cells and non-contextual embeddings (Devlin et al., 2019). In our work, we enriched both architectures. We modified the architecture of the LSTM neural network with fastText embeddings and BERT-based model by adding additional morphological information.

3.2 Adding universal part-of-speech tags

Part-of-speech (POS) is a category of words with similar grammatical properties. POS tags are labels of different word categories, e.g., nouns, verbs, adverbs, adjectives, etc. Universal part-of-speech tags (uPOS tags) aim to standardise part-of-speech tagging (a process of assigning POS tags) across different languages. The uPOS tags we used, were obtained by automatically annotating each word with models trained specifically for this task (Qi et al., 2019).

To the baseline LSTM network with fastText embeddings, as described above, we added uPos tags as shown in Figure 3. We used up to 128 words per sentence. We concatenated the output of the LSTM layer with shape of 128 X 256 with uPOS matrix of shape 128 X 15. We used this 128 X 271 matrix as the input to two additional feed-forward layers of size 256 and the final classification layer. During training, the fastText encoding was not modified, i.e. the corresponding layer was frozen.

Figure 3 illustrates adding uPos tags to the LSTM or BERT model. As an input to BERT, we used maximally 128 tokens (number of byte-pair encoding segments in a sentence). All tokens beyond that were discarded. BERT penultimate layer has a shape of 128 X 768 (disregarding batches). We concatenated this matrix with another one of shape 128 X 15, that contained data about uPOS tags. uPOS matrix was created by embedding class of each uPOS word to a vector of length 15. If a word was split into multiple tokens, we used the same uPOS embedding for each part. Concatenated matrix of size 128 X 783 was than passed to two additional feed-forward layers of size 768 and the final classification layer, as visualised in Figure 3. We used this architecture for fine-tuning all layers of the model on NER datasets in three morphologically rich languages: Slovene, Estonian and Finnish.

3.3 Adding universal features

Universal features contain information about additional grammatical and lexical properties of words, not covered by uPOS tags, e.g., a grammatical tense, number, declination, conjugation, etc. We covered 23 additional features. Each word only has one uPOS tag, but may have multiple additional features. We added these features by concatenating them with uPOS tags and the corresponing architecture, as shown in Figure 4. Each feature class was embedded to a vector of length 15. When concatenating additional data with for example LSTM, the concatenated matrix consisted of LSTM layer output (128 X 256), uPOS matrix (128 X 15) and 23 additional features each represented by another 128 X 15 matrix, making it an overall shape of 128 X 616. The BERT-based model was extended in the same way. Again, this architecture was tested on three morphologically-rich languages: Slovene, Estonian and Finnish. The additional features were obtained from neural models specifically trained for the task of universal features (Qi et al., 2019).

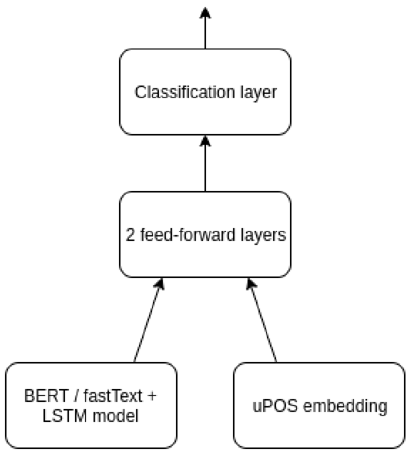

Figure 3: Neural network architecture based on either BERT multilingual model or LSTM model with fastText embeddings. The model includes uPOS tags.

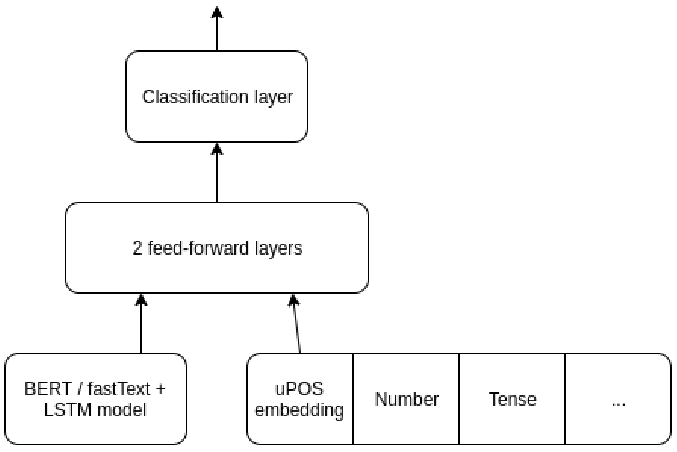

Figure 4: Neural network architecture based on either BERT multilingual model or LSTM model with fastText embeddings. The architecture includes uPOS tags and universal features.

3.4 Adding prefixes and suffixes

For many morphologically-rich languages, including Slavic, Baltic and Finnish, the majority of morphological information is contained in prefixes and suffixes of the words. To test if explicit inclusion of this information can help in neural network classifiers, we obtained lists of prefixes and suffixes for Slovene. We iterated over all words in sentences and checked if they began with a prefix or ended with a suffix on the list. We used this information in our input to neural networks in the same way as uPOS tags and universal features. The problem with this approach is that a word may begin with the letters that make up a prefix or end with letters that make up a suffix, without these letters actually forming a prefix or suffix (e.g., in Slovene ”pri” is a frequent prefix, but does not play a role of prefix in the word ”prizma”). We added prefixes and suffixes in the same way as uPOS and universal features. The only difference was that instead of embedding prefixes and suffixes into embedding of length 15, we embedded them into the vector of length 30. The reason for this are more possible classes to which words might belong (78 prefixes and 331 suffixes).

4 Initial evaluation

We evaluated the proposed architectures with injected additional morphological information using the NER task on Slovene, Estonian, and Finnish. We first shortly repeat the description of the NER data set from *D1.1 Datasets, benchmarks and evaluation metrics for cross-lingual word embeddings*, followed by the results.

4.1 The NER dataset

We test the proposed architectures on a popular downstream task of Named Entity Recognition (NER). NER is an information extraction task that seeks to locate and classify named entity mentions in unstructured text into pre-defined categories such as the person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc.

We obtained labelled datasets for all EMBEDDIA languages. The details of these datasets are presented in Table 1. The NER datasets for EMBEDDIA languages in Table 1 vary in the used label sets, some using more specific labels than others, like job, nicknames etc.

The labels of the NER datasets used in this report are simplified to a common label set of three labels, present in all the addressed working languages, i.e. there are only three labels (*LOC*, *ORG*, and *PER*) in the intersection of all label sets. Due to this diversity in annotations and to make comparison sensible across languages, we trimmed labels in all datasets down to these three classes. Each word in NER datasets is annotated with either named entity label or *OTHR* (the tag used for all words that are not annotated named entities).

Table 1: The collected datasets for NER task and their properties: number of sentences, number of tagged words, availability, and link to the corpus location.

Language	Corpus	Sentences	Tags	Avail.	Location
Croatian	hr500k	25000	29000	public	link
English	CoNLL-2003 NER	21000	44000	public	link
Estonian	Estonian NER corpus	14000	21000	public	link
Finnish	FiNER data	14500	17000	public	link
Latvian	LV Tagger train data	10000	11500	public	link
Lithuanian	TildeNER	5500	7000	limited	link
Russian	factRuEval-2016	5000	9500	public	link
Slovene[2]	ssj500k	9500	9500	public	link
Swedish	Swedish NER	8500	7500	public	link

4.2 Results of adding explicit morphological information to neural networks on the NER task

We compared the baseline models, described in Section 3.1, with LSTM and BERT-based models injected with additional morphological information, described in Sections 3.2, 3.3, and 3.4. As the process of fine-tuning BERT-based models with additional information or to a specific task requires considerable computational resources (GPU power), we limited the number of experiments and languages. Nevertheless, we believe that the results can be generalised to all EMBEDDIA languages as we are using the multilingual BERT model as a baseline that is trained on all the EMBEDDIA languages.

As the evaluation metrics we used precision, recall and F_1 score. We calculated each metric for each named entity class (ORG, LOC, and PER) and calculated weighted average per metric to produce the

[2]The Slovene ssj500k originally contains more sentences, but only 9500 are annotated with NER data.

Table 2: An example of calculating the weighted results. The weighted precision p_w is calculated as $p_w = (p_{ORG} \cdot n_{ORG} + p_{LOC} \cdot n_{LOC} + p_{PER} \cdot n_{PER}/(n_{ORG} + n_{LOC} + n_{PER})$, where p_{label} is precision of a given label (ORG, LOC, or PER), and n_{label} is the number of instances with the given label. In our example, p_w is calculated as $p_w = (0.6572 * 358 + 0.8387 * 115 + 0.7360 * 227)/700 = 0.7126$.

Label	Precision	Recall	F_1 score	n_{label}
ORG	0.6572	0.7123	0.6836	358
LOC	0.8387	0.9043	0.8703	115
PER	0.7360	0.8106	0.7715	227
weighted / total	0.7126	0.7757	0.7428	700

final results, which are shown in the results tables. The weighted averages use frequencies of class values in each dataset as the weights. An example of calculated results is shown in Table 2. Note that the we followed the standard approach in NER evaluation and ignored the tag OTHR, which was used for all words that are not annotated named entities, i.e. we only focused on the named entities. All models were evaluated using 11-fold cross-validation[3].

For each of the three analyzed morphologically rich languages (Slovene, Estonian, and Finnish), we trained eight different models on the NER dataset.

- *fastText + LSTM* is a LSTM network that uses fastText non-contextual embeddings.
- *fastText + LSTM + 2FFL* is a LSTM network, similar to *fastText + LSTM* with additional two feed forward layers at the end.
- *fastText + LSTM + 2FFL + uPOS* models uses architecture similar to *fastText + LSTM + 2FFL* with additional information about uPOS tags.
- *fastText + LSTM + 2FFL + uPOS + feats* uses information about feats as well as uPOS.
- *BERT* is a neural network model that takes weights from the BERT multilingual model and is fine-tuned for the NER task.
- *BERT + uPOS* use BERT multilingual model and uPOS tags. It differs from *BERT* model by also utilising two feed-forward layers.
- *BERT + uPOS + feats* models uses BERT multilingual model, uPOS tags, and universal features.
- *BERT + uPOS + feats + fixes* uses explicit information on prefixes and suffixes within the model using uPOS tags and universal features. Note, that due to availability of prefixes and suffixes this model was trained only for the Slovenian language.

For each model and language, we used 10 epochs of fine-tuning BERT for NER task and 50 epochs for training the LSTM with fastText We have chosen the parameters for our training in a preliminary testing on a Slovene dataset with the aim to balance the performance and computation times. The results for Slovene, Estonian, and Finnish are presented in Tables 3, 4, 5, respectively.

We confirmed our hypothesis that additional morphological information can improve results of non-contextual embeddings, as *fastText + LSTM + 2FFL + uPOS + feats* model outperformed *fastText + LSTM* across multiple languages. We included model *fastText + LSTM + 2FFL* to test whether improvements might be a result of two additional feed forward layers, but this was not the case. The results show that LSTM networks with fastText embeddings are not competitive with BERT-based models in none of the tested languages, which confirms the advantage of using contextual embeddings.

In general, adding uPOS tags, universal features, and prefixes/suffixes does not improve the results over the baseline BERT model (though, the results are slightly improved for the Sovenian language). To confirm this, we performed the t-test on cross-validation folds of different models. The test results

[3]The unusual choice of 11- instead of more common 10-fold cross-validation is a consequence of a typo in the initial evaluation script and does affect the results.

Table 3: Results for different models on NER task in the Slovene language. The result of the best model for each measure is in bold.

Model	Precision	Recall	F_1 score
fastText + LSTM	0.6494	0.6321	0.6389
fastText + LSTM + 2FFL	0.6393	0.6664	0.6509
fastText + LSTM + 2FFL + uPOS	0.6712	0.6900	0.6785
fastText + LSTM + 2FFL + uPOS + feats	0.6594	0.6797	0.6657
BERT	0.8332	0.8516	0.8415
BERT + uPOS	0.8318	0.8495	0.8400
BERT + uPOS + feats	**0.8411**	**0.8526**	**0.8458**
BERT + uPOS + feats + fixes	0.8372	**0.8526**	0.8438

Table 4: Results for different models on NER task in the Estonian language. The result of the best model for each measure is in bold.

Model	Precision	Recall	F_1 score
fastText + LSTM	0.7089	0.7170	0.7108
fastText + LSTM + 2FFL	0.7062	0.7225	0.7108
fastText + LSTM + 2FFL + uPOS	0.7045	0.7391	0.7192
fastText + LSTM + 2FFL + uPOS + feats	0.7074	0.7422	0.7222
BERT	**0.8713**	**0.8829**	**0.8765**
BERT + uPOS	0.8681	0.8822	0.8746
BERT + uPOS + feats	0.8680	0.8760	0.8715

Table 5: Results for different models on NER task in the Finnish language. The result of the best model for each measure is in bold.

Model	Precision	Recall	F_1 score
fastText + LSTM	0.7706	0.8027	0.7855
fastText + LSTM + 2FFL	0.7642	0.8304	0.7948
fastText + LSTM + 2FFL + uPOS	0.7833	0.8327	0.8062
fastText + LSTM + 2FFL + uPOS + feats	0.7917	0.8256	0.8071
BERT	**0.9242**	**0.9261**	**0.9247**
BERT + uPOS	0.9192	0.9183	0.9183
BERT + uPOS + feats	0.9184	0.9197	0.9186

show that we cannot reject the null hypothesis of identical average scores between all BERT based models, but we can reject (at $p = 0.01$) the hypothesis of equal performance of BERT models and LSTM models with fastText embeddings. We hypothesise that BERT-based models have already captured the information about the morphology of the words.

5 Prediction Uncertainty Estimation

One of the shortcomings of standard deep neural networks is that they do not provide information on reliability of predictions. Recent works on combining probabilistic Bayesian inference and neural network methodology attracted much attention in the scientific community (Myshkov & Julier, 2016). The main reason is the ability of probabilistic neural networks to quantify trustworthiness of predicted results. This information can be important, especially in tasks were decision making plays an important role (Miok, 2018).

The areas which can significantly benefit from prediction uncertainty estimation are text classification tasks which trigger specific actions. In the context of the EMBEDDIA project, this includes hate speech detection in user generated contents, frequently encountered in the media industry. Here, reliable results are needed to remove harmful contents and possibly ban malicious users without preventing the freedom of speech. In order to assess the uncertainty of the predicted values, the neural networks require a Bayesian framework.

Bayesian neural network (BNN) methodology provide reliability scores by probabilistic interpretation of model parameters. Apart from prediction uncertainty estimation, BNNs offer robustness to overfitting and can be efficiently trained on small data sets (Kucukelbir et al., 2017). However, neural networks that apply Bayesian inference can be computationally expensive, especially the ones with the complex, deep architectures. Our work is based on Monte Carlo Dropout (MCD) method proposed by Gal & Ghahramani (2016). The idea of this approach is to capture prediction uncertainty using the dropout as a regularization technique (Srivastava et al., 2014). The approach drops some randomly selected nodes from the neural network during the training process. Dropout increases the robustness of networks and prevents overfitting. Different variants of dropout improved classification results in various areas (Baldi & Sadowski, 2013). Gal & Ghahramani (2016) exploited the interpretation of dropout as a Bayesian approximation and proposed a Monte Carlo dropout (MCD) approach to estimate the prediction uncertainty.

In this work, we analyze the applicability of Monte Carlo dropout in assessing the predictive uncertainty. While the complete description of the proposed approach is contained in the appended paper by Miok et al. (2019), published in the Proceedings of the Statistical Language and Speech Processing (SLSP) 2019 conference, this section provides a brief overview of this work. Our main goal was to propose a novel methodology for reliability assessment of neural classifiers' predictions on textual data, and test it on different hate speech datasets. For a given text we provide a probabilistic assessment of the prediction uncertainty and illustrate it in a comprehensible visual form. The outline of the used methodology is presented in Figure 5.

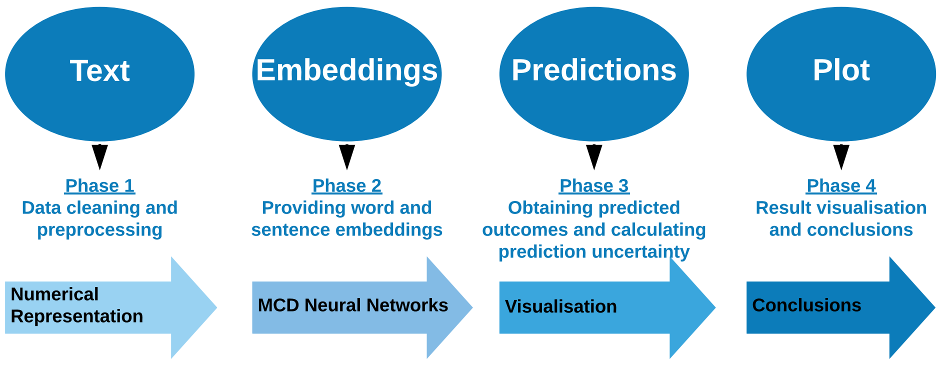
Text
Embeddings
Predictions
Plot
Phase 1
Data cleaning and preprocessing
Phase 2
Providing word and sentence embeddings
Phase 3
Obtaining predicted outcomes and calculating prediction uncertainty
Phase 4
Result visualisation and conclusions
Numerical Representation
MCD Neural Networks
Visualisation
Conclusions

Figure 5: The diagram of the proposed methodology.

In the prediction phase, we used LSTM models with Monte Carlo dropout for reliability assesmnet and showed that in hate speech classification their performance is comparable to the best competing approaches using word embeddings and superior using sentence embeddings. Our study also showed that pre-trained sentence embeddings outperform even state-of-the-art contextual word embeddings and can be recommended as a suitable representation for this task.

The main contributions of the work are:

- investigation of prediction uncertainty assessment in the area of text classification,
- implementation of hate speech detection with reliability output,
- evaluation of different contextual embedding approaches in the area of hate speech,
- a novel visualisation of prediction uncertainty and errors of classification models.

6 Associated outputs

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

Description	URL	Availability
Morphological BERT	`github.com/EMBEDDIA/morphological-BERT`	To become public*
Morphological fastText	`github.com/EMBEDDIA/morphological-fasttext`	To become public*
Hate speech prediction	`github.com/EMBEDDIA/Hate-Speech-Prediction-Uncertainty`	Public (MIT)

* Resources marked here as "To become public" are available only within the consortium while under development and/or associated with work yet to be published. They will be released publicly when the associated work is completed and published.

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

Citation	Status	Appendix
Miok, K., Nguyen-Doan, D., Škrlj, B., Zaharie, D., Robnik-Šikonja, M. (2019). Prediction uncertainty estimation for hate speech classification. In Proceedings of the international conference on statistical language and speech processing SLSP 2019 (pp. 286–298). Springer.	Published	Appendix A

7 Conclusions and further work

Most of the standard neural network architectures for language processing are adapted to English. As morphologically rich languages might require different inputs and explicit information about morphology, we tested adding such information to i) the standard deep neural network approach with LSTM cells using non-contextual fast embeddings, and ii) the state-of-the-art contextual multilingual model BERT. Both models were combined with additional morphological information: separately trained universal POS tags, separately trained universal features, as well as prefixes and suffixes. We tested the modified architectures on the named entity recognition problem using three morphologically rich languages: Slovene, Estonian, and Finnish.

We showed that combining non-contextual embeddings with additional morphological information did improve the results over the baseline LSTM model with fastText. However, usage of BERT multilingual model together with universal part-of-speech tags, universal features, and prefixes/suffixes has not shown any advantage over the baseline BERT model. We believe that the baseline BERT model already captures all the necessary morphological information for the NER task.

We presented the first successful approach to assessment of prediction uncertainty in hate speech classification. Our approach uses LSTM model with Monte Carlo dropout and shows performance comparable to the best competing approaches using word embeddings and superior performance using sentence embeddings. Our study shows that pre-trained sentence embeddings outperform even state-of-the-art contextual word embeddings and can be recommended as a suitable representation for this task. We demonstrate that reliability of predictions and errors of the models can be comprehensively visualized.

We see a possible path to improvements in the performance of a classifier by improving the BERT model, in particular in its training on larger corpora specific to a given language (at the moment we are using a pre-trained BERT multilingual model, trained on Wikipedia corpora from 104 languages). We have already taken the first steps in this direction, but due to long training times of BERT models (over a month for each language), the results are not yet available at the time of this report. Other possible improvements include adding additional morphological information to the attention mechanism of BERT while training the baseline models instead of concatenating this data with pre-trained models. We could also introduce additional language information to neural networks, in particular dependency relations.

Prediction uncertainty estimation is rarely implemented for text classification and other NLP tasks, hence our future work will also go in this direction. While cross-lingual embeddings possibly open new opportunities to share data sets and models between languages, the evaluation in rare languages is difficult, therefore the assessment of predictive reliability for such problems might be an auxiliary evaluation approach. In this context, we also plan to investigate convolutional neural networks and transformer networks with probabilistic interpretation.

References

Baldi, P., & Sadowski, P. J. (2013). Understanding dropout. In *Advances in neural information processing systems* (pp. 2814–2822).

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

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). 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).

Gal, Y., & Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International conference on machine learning* (pp. 1050–1059).

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Kim, Y., Jernite, Y., Sontag, D., & Rush, A. M. (2016). Character-aware neural language models. In *Aaai* (pp. 2741–2749).

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., & Blei, D. M. (2017). Automatic differentiation variational inference. *The Journal of Machine Learning Research*, *18*(1), 430–474.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

Mikolov, T., Le, Q. V., & Sutskever, I. (2013). Exploiting similarities among languages for machine translation. *arXiv preprint 1309.4168*.

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

Miok, K. (2018, 09). Estimation of prediction intervals in neural network-based regression models. In *20th international symposium on symbolic and numeric algorithms for scientific computing (synasc)* (p. 463-468).

Miok, K., Nguyen-Doan, D., Škrlj, B., Zaharie, D., & Robnik-Šikonja, M. (2019). Prediction uncertainty estimation for hate speech classification. In *International conference on statistical language and speech processing* (pp. 286–298).

Myshkov, P., & Julier, S. (2016). Posterior distribution analysis for bayesian inference in neural networks. In *Workshop on bayesian deep learning, nips.*

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).

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

Pires, T., Schlinger, E., & Garrette, D. (2019). How multilingual is multilingual bert? *arXiv preprint arXiv:1906.01502*.

Qi, P., Dozat, T., Zhang, Y., & Manning, C. D. (2019). Universal dependency parsing from scratch. *arXiv preprint arXiv:1901.10457*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

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

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., & Bowman, S. (2018). Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 emnlp workshop blackboxnlp: Analyzing and interpreting neural networks for nlp* (pp. 353–355).

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems* (pp. 649–657).

Prediction Uncertainty Estimation for Hate Speech Classification

Kristian Miok[1], Dong Nguyen-Doan[1], Blaž Škrlj[2], Daniela Zaharie[1], and Marko Robnik-Šikonja[3]

[1] Computer Science Department, West University of Timisoara,
Bulevardul Vasile Pârvan 4, 300223 Timișoara, Romania
{kristian.miok,dong.nguyen10,daniela.zaharie}@e-uvt.ro
[2]Jožef Stefan Institute and Jožef Stefan International Postgraduate School,
Jamova 39, 1000 Ljubljana, Slovenia
blaz.skrlj@ijs.si
[3] Faculty of Computer and Information Science, University of Ljubljana,
Večna pot 113, 1000 Ljubljana, Slovenia
marko.robnik@fri.uni-lj.si

The final authenticated publication is available online at SLSP 2019 conference link: `https://doi.org/10.1007/978-3-030-31372-2_24`.

Abstract. As a result of social network popularity, in recent years, hate speech phenomenon has significantly increased. Due to its harmful effect on minority groups as well as on large communities, there is a pressing need for hate speech detection and filtering. However, automatic approaches shall not jeopardize free speech, so they shall accompany their decisions with explanations and assessment of uncertainty. Thus, there is a need for predictive machine learning models that not only detect hate speech but also help users understand when texts cross the line and become unacceptable.
The reliability of predictions is usually not addressed in text classification. We fill this gap by proposing the adaptation of deep neural networks that can efficiently estimate prediction uncertainty. To reliably detect hate speech, we use Monte Carlo dropout regularization, which mimics Bayesian inference within neural networks. We evaluate our approach using different text embedding methods. We visualize the reliability of results with a novel technique that aids in understanding the classification reliability and errors.

Keywords: prediction uncertainty estimation, hate speech classification, Monte Carlo dropout method, visualization of classification errors

1 Introduction

Hate speech represents written or oral communication that in any way discredits a person or a group based on characteristics such as race, color, ethnicity, gender, sexual orientation, nationality, or religion [35]. Hate speech targets disadvantaged social groups and harms them both directly and indirectly [33]. Social networks like Twitter and

Facebook, where hate speech frequently occurs, receive many critics for not doing enough to deal with it. As the connection between hate speech and the actual hate crimes is high [4], the importance of detecting and managing hate speech is not questionable. Early identification of users who promote such kind of communication can prevent an escalation from speech to action. However, automatic hate speech detection is difficult, especially when the text does not contain explicit hate speech keywords. Lexical detection methods tend to have low precision because, during classification, they do not take into account the contextual information those messages carry [11]. Recently, contextual word and sentence embedding methods capture semantic and syntactic relation among the words and improve prediction accuracy.

Recent works on combining probabilistic Bayesian inference and neural network methodology attracted much attention in the scientific community [23]. The main reason is the ability of probabilistic neural networks to quantify trustworthiness of predicted results. This information can be important, especially in tasks were decision making plays an important role [22]. The areas which can significantly benefit from prediction uncertainty estimation are text classification tasks which trigger specific actions. Hate speech detection is an example of a task where reliable results are needed to remove harmful contents and possibly ban malicious users without preventing the freedom of speech. In order to assess the uncertainty of the predicted values, the neural networks require a Bayesian framework. On the other hand, Srivastava et al. [32] proposed a regularization approach, called dropout, which has a considerable impact on the generalization ability of neural networks. The approach drops some randomly selected nodes from the neural network during the training process. Dropout increases the robustness of networks and prevents overfitting. Different variants of dropout improved classification results in various areas [1]. Gal and Ghahramani [14] exploited the interpretation of dropout as a Bayesian approximation and proposed a Monte Carlo dropout (MCD) approach to estimate the prediction uncertainty. In this paper, we analyze the applicability of Monte Carlo dropout in assessing the predictive uncertainty.

Our main goal is to accurately and reliably classify different forms of text as hate or non-hate speech, giving a probabilistic assessment of the prediction uncertainty in a comprehensible visual form. We also investigate the ability of deep neural network methods to provide good prediction accuracy on small textual data sets. The outline of the proposed methodology is presented in Figure 1.

Our main contributions are:

- investigation of prediction uncertainty assessment to the area of text classification,
- implementation of hate speech detection with reliability output,
- evaluation of different contextual embedding approaches in the area of hate speech,
- a novel visualization of prediction uncertainty and errors of classification models.

The paper consists of six sections. In Section 2, we present related works on hate speech detection, prediction uncertainty assessment in text classification context, and visualization of uncertainty. In Section 3, we propose the methodology for uncertainty assessment using dropout within neural network models, as well as our novel visualization of prediction uncertainty. Section 4 presents the data sets and the experimental scenario. We discuss the obtained results in Section 5 and present conclusions and ideas for further work in Section 6.

2 Related Work

We shortly present the related work in three areas which constitute the core of our approach: hate speech detection, recurrent neural networks with Monte Carlo dropout for assessment of prediction uncertainty in text classification, and visualization of predictive uncertainty.

2.1 Hate Speech Detection

Techniques used for hate speech detection are mostly based on supervised learning. The most frequently used classifier is the Support Vector Machines (SVM) method [30]. Recently, deep neural networks, especially recurrent neural network language models [20], became very popular. Recent studies compare (deep) neural networks [28,9,12] with the classical machine learning methods.

Our experiments investigate embeddings and neural network architectures that can achieve superior predictive performance to SVM or logistic regression models. More specifically, our interest is to explore the performance of MCD neural networks applied to the hate speech detection task.

2.2 Prediction Uncertainty in Text Classification

Recurrent neural networks (RNNs) are a popular choice in text mining. The dropout technique was first introduced to RNNs in 2013 [34] but further research revealed negative impact of dropout in RNNs, especially within language modeling. For example, the dropout in RNNs employed on a handwriting recognition task, disrupted the ability of recurrent layers to effectively model sequences [25]. The dropout was successfully applied to language modeling by [36] who applied it only on fully connected layers. The then state-of-the-art results were explained with the fact that by using the dropout, much deeper neural networks can be constructed without danger of overfitting. Gal and Ghahramani [15] implemented the variational inference based dropout which can also regularize recurrent layers. Additionally, they provide a solution for dropout within word embeddings. The method mimics Bayesian inference by combining probabilistic parameter interpretation and deep RNNs. Authors introduce the idea of augmenting probabilistic

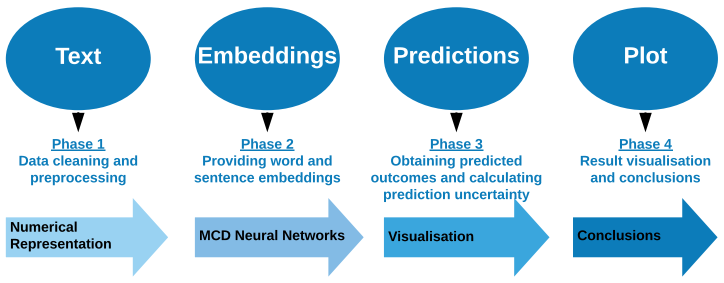
Text
Embeddings
Predictions
Plot
Phase 1
Data cleaning and preprocessing
Phase 2
Providing word and sentence embeddings
Phase 3
Obtaining predicted outcomes and calculating prediction uncertainty
Phase 4
Result visualisation and conclusions
Numerical Representation
MCD Neural Networks
Visualisation
Conclusions

Fig. 1. The diagram of the proposed methodology.

RNN models with the prediction uncertainty estimation. Recent works further investigate how to estimate prediction uncertainty within different data frameworks using RNNs [37]. Some of the first investigation of probabilistic properties of SVM prediction is described in the work of Platt [26]. Also, investigation how Bayes by Backprop (BBB) method can be applied to RNNs was done by [13].

Our work combines the existing MCD methodology with the latest contextual embedding techniques and applies them to hate speech classification task. The aim is to obtain high quality predictions coupled with reliability scores as means to understand the circumstances of hate speech.

2.3 Prediction Uncertainty Visualization in Text Classification

Visualizations help humans in making decisions, e.g., select a driving route, evacuate before a hurricane strikes, or identify optimal methods for allocating business resources. One of the first attempts to obtain and visualize latent space of predicted outcomes was the work of Berger et al. [2]. Prediction values were also visualized in geo-spatial research on hurricane tracks [10,29]. Importance of visualization for prediction uncertainty estimation in the context of decision making was discussed in [18,17].

We are not aware of any work on prediction uncertainty visualization for text classification or hate speech detection. We present visualization of tweets in a two dimensional latent space that can reveal relationship between analyzed texts.

3 Deep Learning with Uncertainty Assessment

Deep learning received significant attention in both NLP and other machine learning applications. However, standard deep neural networks do not provide information on reliability of predictions. Bayesian neural network (BNN) methodology can overcome this issue by probabilistic interpretation of model parameters. Apart from prediction uncertainty estimation, BNNs offer robustness to overfitting and can be efficiently trained on small data sets [16]. However, neural networks that apply Bayesian inference can be computationally expensive, especially the ones with the complex, deep architectures. Our work is based on Monte Carlo Dropout (MCD) method proposed by [14]. The idea of this approach is to capture prediction uncertainty using the dropout as a regularization technique.

In contrast to classical RNNs, Long Short-term Memory (LSTM) neural networks introduce additional gates within the neural units. There are two sources of information for specific instance t that flows through all the gates: input values x_t and recurrent values that come from the previous instance h_{t-1}. Initial attempts to introduce dropout within the recurrent connections were not successful, reporting that dropout brakes the correlation among the input values. Gal and Ghahramani [15] solve this issue using predefined dropout mask which is the same at each time step. This opens the possibility to perform dropout during each forward pass through the LSTM network, estimating the whole distribution for each of the parameters. Parameters' posterior distributions that are approximated with such a network structure, $q(\omega)$, is used in constructing posterior predictive distribution of new instances y^*:

$$p(y^*|x^*, D) \approx \int p\big(y^*|f^{\omega}(x^*)\big)\; q(\omega) d\omega, \tag{1}$$

where $p\big(y^*|f^{\omega}(x^*)\big)$ denotes the likelihood function. In the regression tasks, this probability is summarized by reporting the means and standard deviations while for classification tasks the mean probability is calculated as:

$$\frac{1}{K}\sum_{k=1}^{K} p(y^*|x^*, \hat{\omega}_k) \tag{2}$$

where $\hat{\omega}_k \sim q(\omega)$. Thus, collecting information in K dropout passes throughout the network during the training phase is used in the testing phase to generate (sample) K predicted values for each of the test instance. The benefit of such results is not only to obtain more accurate prediction estimations but also the possibility to visualize the test instances within the generated outcome space.

3.1 Prediction Uncertainty Visualization

For each test instance, the neural network outputs a vector of probability estimates corresponding to the samples generated through Monte Carlo dropout. This creates an opportunity to visualize the variability of individual predictions. With the proposed visualization, we show the correctness and reliability of individual predictions, including false positive results that can be just as informative as correctly predicted ones. The creation of visualizations consists of the following five steps, elaborated below.

1. Projection of the vector of probability estimates into a two dimensional vector space.
2. Point coloring according to the mean probabilities computed by the network.
3. Determining point shapes based on correctness of individual predictions (four possible shapes).
4. Labeling points with respect to individual documents.
5. Kernel density estimation of the projected space — this step attempts to summarize the instance-level samples obtained by the MCD neural network.

As the MCD neural network produces hundreds of probability samples for each target instance, it is not feasible to directly visualize such a multi-dimensional space. To solve this, we leverage the recently introduced UMAP algorithm [19], which projects the input d dimensional data into a s-dimensional (in our case $s = 2$) representation by using computational insights from the manifold theory. The result of this step is a two dimensional matrix, where each of the two dimensions represents a latent dimension into which the input samples were projected, and each row represents a text document.

In the next step, we overlay the obtained representation with other relevant information, obtained during sampling. Individual points (documents) are assigned the mean probabilities of samples, thus representing the reliability of individual predictions. We discretize the $[0, 1]$ probability interval into four bins of equal size for readability purposes. Next, we shape individual points according to the correctness of predictions. We take into account four possible outcomes (TP - true positives, FP - false positives, TN - true negatives, FN - false negatives).

As the obtained two dimensional projection represents an approximation of the initial sample space, we compute the kernel density estimation in this subspace and thereby outline the main neural network's predictions. We use two dimensional Gaussian kernels for this task.

The obtained estimations are plotted alongside individual predictions and represent densities of the neural network's focus, which can be inspected from the point of view of correctness and reliability.

4 Experimental Setting

We first present the data sets used for the evaluation of the proposed approach, followed by the experimental scenario. The results are presented in Section 5.

4.1 Hate Speech Data Sets

We use three data sets related to the hate speech.

1 - HatEval data set is taken from the SemEval task "Multilingual detection of hate speech against immigrants and women in Twitter (hatEval)[1]". The competition was organized for two languages, Spanish and English; we only processed the English data set. The data set consists of 100 tweets labeled as 1 (hate speech) or 0 (not hate speech).

2 - YouToxic data set is a manually labeled text toxicity data, originally containing 1000 comments crawled from YouTube videos about the Ferguson unrest in 2014[2]. Apart from the main label describing if the comment is hate speech, there are several other labels characterizing each comment, e.g., if it is a threat, provocative, racist, sexist, etc. (not used in our study). There are 138 comments labeled as a hate speech and 862 as non-hate speech. We produced a data set of 300 comments using all 138 hate speech comments and randomly sampled 162 non-hate speech comments.

3 - OffensiveTweets data set[3] originates in a study regarding hate speech detection and the problem of offensive language [11]. Our data set consists of 3000 tweets. We took 1430 tweets labeled as hate speech and randomly sampled 1670 tweets from the collection of remaining 23 353 tweets.

[1] https://competitions.codalab.org/competitions/19935

[2] https://zenodo.org/record/2586669#.XJiS8ChKi70

[3] https://github.com/t-davidson/hate-speech-and-offensive-language

Data Preprocessing Social media text use specific language and contain syntactic and grammar errors. Hence, in order to get correct and clean text data we applied different prepossessing techniques without removing text documents based on the length. The pipeline for cleaning the data was as follows:

- Noise removal: user-names, email address, multiple dots, and hyper-links are considered irrelevant and are removed.
- Common typos are corrected and typical contractions and hash-tags are expanded.
- Stop words are removed and the words are lemmatized.

4.2 Experimental Scenario

We use logistic regression (LR) and Support Vector Machines (SVM) from the scikit-learn library [5] as the baseline classification models. As a baseline RNN, the LSTM network from the Keras library was applied [8]. Both LSTM and MCD LSTM networks consist of an embedding layer, LSTM layer, and a fully connected layer within the Word2Vec and ELMo embeddings. The embedding layer was not used in TF-IDF and Universal Sentence encoding.

To tune the parameters of LR (i.e. *liblinear* and *lbfgs* for the solver functions and the number of component C from 0.01 to 100) and SVM (i.e. the *rbf* for the kernel function, the number of components C from 0.01 to 100 and the gamma γ values from 0.01 to 100), we utilized the random search approach [3] implemented in scikit-learn. In order to obtain best architectures for the LSTM and MCD LSTM models, various number of units, batch size, dropout rates and so on were fine-tuned.

5 Evaluation and Results

We first describe experiments comparing different word representations, followed by sentence embeddings, and finally the visualization of predictive uncertainty.

5.1 Word Embedding

In the first set of experiments, we represented the text with word embeddings (sparse TF-IDF [31] or dense word2vec [21], and ELMo [24]). We utilise the gensim library [27] for word2vec model, the scikit-learn for TFIDF, and the ELMo pretrained model from TensorFlow Hub[4]. We compared different classification models using these word embeddings. The results are presented in Table 1.

The architecture of LSTM and MCD LSTM neural networks contains an embedding layer, LSTM layer, and fully-connected layer (i.e. dense layer) for word2vec and ELMo word embeddings. In LSTM, the recurrent dropout is applied to the units for linear transformation of the recurrent state and the classical dropout is used for the units with the linear transformation of the inputs. The number of units, recurrent dropout, and dropout probabilities for LSTM layer were obtained by fine-tuning (i.e. we used 512, 0.2 and 0.5 for word2vec and TF-IDF, 1024, 0.5, and 0.2 for ELMo in the experiments with

[4] https://tfhub.dev/google/elmo/2

MCD LSTM architecture). The search ranges for hyper parameter tuning are described in Table 2.

Table 1. Comparison of classification accuracy (with standard deviation in brackets) for word embeddings, computed using 5-fold cross-validation. All the results are expressed in percentages and the best ones for each data set are in bold.

	HatEval			**YouToxic**			**OffensiveTweets**		
Model	**TF-IDF**	**W2V**	**ELMo**	**TF-IDF**	**W2V**	**ELMo**	**TF-IDF**	**W2V**	**ELMo**
Logistic Regression	68.0 [2.4]	54.0 [13.6]	62.0 [6.8]	69.3 [3.0]	54.0 [3.0]	**76.6 [6.1]**	**77.2 [1.1]**	68.0 [2.4]	75.6 [1.2]
SVM	63.0 [5.1]	66.0 [3.7]	62.0 [12.9]	70.6 [4.2]	55.0 [3.4]	73.3 [5.5]	77.0 [0.7]	59.6 [1.5]	73.0 [1.9]
LSTM	69.0 [7.3]	67.0 [6.8]	66.0 [12.4]	66.6 [2.3]	59.3 [4.6]	74.3 [2.7]	73.4 [0.8]	75.0 [1.7]	74.7 [1.9]
MCD LSTM	67.0 [10.8]	**69.0 [6.6]**	67.0 [9.8]	66.0 [3.7]	59.3 [3.8]	75.3 [5.5]	71.1 [1.6]	72.0 [1.6]	75.2 [0.9]

Table 2. Hyper-parameters for LSTM and MCD LSTM models

Name	**Parameter type**	**Values**
Optimizers	Categorical	Adam, rmsprop
Batch size	Discrete	4 to 128, step=4
Activation function	Categorical	tanh, relu and linear
Number of epochs	Discrete	10 to 100, step=5
Number of units	Discrete	128, 256, 512, or 1024
Dropout rate	Float	0.1 to 0.8, step=0.05

The classification accuracy for HatEval data set is reported in the Table 1 (left). The difference between logistic regression and the two LSTM models indicates accuracy improvement once the recurrent layers are introduced. On the other hand, as the ELMo embedding already uses the LSTM layer to take into account semantic relationship among the words, no notable difference between logistic regression and LSTM models can be observed using this embedding.

Results for YouToxic and OffensiveTweets data sets are presented in Table 1 (middle) and (right), respectively. Similarly to the HatEval data set, there is a difference between the logistic regression and the two LSTM models using the word2vec embeddings. For all data sets, the results with ELMo embeddings are similar across the four classifiers.

5.2 Sentence Embedding

In the second set of experiments, we compared different classifiers using sentence embeddings [6] as the representation. Table 3 (left) displays results for HatEval. We can notice improvements in classification accuracy for all classifiers compared to the word embedding representation in Table 1. The best model for this small data set is MCD LSTM. For larger YouToxic and OffensiveTweets data sets, all the models perform comparably. Apart from the prediction accuracy the four models were compared using precision, recall and F1 score [7].

We use the Universal Sentence Encoder module[5] to encode the data. The architecture of LSTM and MCD LSTM contains a LSTM layer and dense layer. With MCD LSTM architecture in the experiments, the number of neurons, recurrent dropout and dropout value for LSTM is 1024, 0.75 and 0.5, respectively. The dense layer has the same number of units as LSTM layer, and the applied dropout rate is 0.5. The hyper-parameters used to tune the LSTM and MCD LSTM models are presented in the Table 2.

Table 3. Comparison of predictive models using sentence embeddings. We present average classification accuracy, precision, recall and F_1 score (and standard deviations), computed using 5-fold cross-validation. All the results are expressed in percentages and the best accuracies are in bold.

	HatEval				**YouToxic**				**OffensiveTweets**			
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**Accuracy**	**Precision**	**Recall**	**F1**	**Accuracy**	**Precision**	**Recall**	**F1**
LR	66.0 [12.4]	67.3 [15.3]	65.2 [15.9]	65.2 [13.1]	77.3 [4.1]	74.3 [7.3]	77.3 [3.6]	75.7 [5.3]	80.8 [1.0]	79.6 [1.9]	84.9 [1.2]	82.2 [1.1]
SVM	67.0 [12.1]	68.2 [15.2]	65.0 [15.8]	65.8 [13.3]	77.3 [6.2]	72.6 [8.6]	80.7 [7.4]	76.3 [7.6]	80.7 [1.3]	78.6 [2.0]	86.7 [1.0]	82.4 [1.2]
LSTM	70.0 [8.4]	70.8 [11.0]	63.1 [17.5]	66.2 [14.4]	76.6 [8.6]	73.4 [11.2]	79.2 [8.0]	75.8 [8.6]	80.7 [1.6]	82.8 [2.1]	79.7 [2.3]	81.1 [1.5]
MCD LSTM	**74.0 [10.7]**	73.4 [12.7]	78.4 [13.6]	74.9 [10.0]	**78.7 [5.8]**	74.7 [9.2]	80.9 [6.5]	77.5 [7.4]	**81.0 [1.2]**	81.5 [1.8]	82.5 [2.7]	81.9 [1.3]

5.3 Visualizing Predictive Uncertainty

In Figure 2 we present a new way of visualizing dependencies among the test tweets. The relations are result of applaing the MCD LSTM network to the HetEval data set. This allows further inspection of the results as well as interpretation of correct and incorrect predictions. To improve comprehensibility of predictions and errors, each point in the visualization is labeled with a unique identifier, making the point tractable to the original document, given in Table 4.

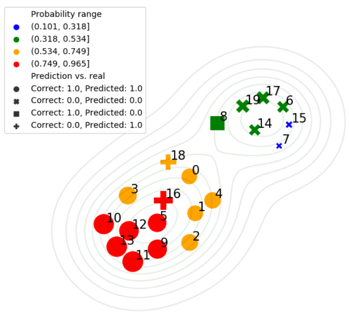

Fig. 2. Visualization of individual predictions along with their probability ranges. The numeric labels besides each point correspond to individual tweets, summarized in Table 4. The size of the points corresponds to the neural network's mean probability for a given prediction. True positives are marked with circles, true negatives with crosses, false positives with squares, and false negatives as pluses.

[5] https://tfhub.dev/google/universal-sentence-encoder-large/3

Table 4. Test documents (one fold) used for the visualization in Figure 2.

Index (see Figure 2)	Corresponding text
0	ok if you fu***** said leave block me. but dm me first i am gonna kick your ass. shut the f*** up you
1	wow mo c*** got hard. want to pull your panties down, push you against these door and let you feel my c*** deep in your wet p****
2	no i said hope. i hope you women learn your place! sitdowninthekitchen
3	so this b**** blocked me but check her pinned tweet out, she is accusing men of rape on tumblr with . . .
4	this is not rape, or anything like it. this a crude pass. this is not 'rape culture.' it is w**** culture.
5	rape is bad but alota h*** be lying bout that s*** women are crazy dawg & will f*** ya life up be careful
6	lol. they did not applaud a rape threat .how long are you going to ride that s***?
7	you had a great chance to show everyone that youre not a c*** on tv tonight and you f***** it up.
8	lmao!!! stuff i say. some stuff be so damn funny.especially when it comes to illegal immigrants who are full fledged criminals coming to america . . .
9	the german government pays for 3 week vacation for refugees to go home -
10	burundian refugees should go home!
11	no benefits to any immigrants !! come here &amp; make it on your own, or go back home !!
12	illegal aliens waiting for first class msc container-shippping in antwerpen stop immigration
13	sick barstewards! this is what happens when we put up the refugees welcome signs! they not only rape our wives or girlfriends . . .
14	i am not forcing anyone. i am just putting out my view. people believe it or not is their prerogative.even if a single person agree with me . . .
15	tajikistani charged with beating new york university professor to death another animal./
16	child prostitution: feds break up mexico-to-queens sex trafficking ring via . . .
17	home office guilty of a serious breach of the duty of candour and cooperation regarding children entitled to enter uk. where did these children go? . . .
18	p.s why do you not pay unemployed people who do endless hours of voluntary work they do that to give something to the community
19	seriously, amy and cindy are bffs, i know that for sure. hmm, mmm.

As Figure 2 shows, the tweets are grouped into two clusters. According to the kernel density isometric lines, two centers are identified: the tweets assigned lower probability of being hate speech and the tweets with higher probability of being hate speech. Let us focus on the wrongly classified tweets and their positions in the graph (tweets 8, 16 and 18). While for tweets 8 and 18 the classifier wasn't certain and a mistake seems possible according to the plot, the tweet 16 was predicted to be hate speech with high probability. Analyzing the words that form this tweet, we notice that not only that most of them often do appear in the hate speech but also this combination of the words used together is very characteristic for the offensive language.

Our short demonstration shows the utility of the proposed visualization which can identify different types of errors and helps to explain weaknesses in the classifier or wrongly labeled data.

6 Conclusions

We present the first successful approach to assessment of prediction uncertainty in hate speech classification. Our approach uses LSTM model with Monte Carlo dropout and shows performance comparable to the best competing approaches using word embeddings and superior performance using sentence embeddings. We demonstrate that reliability of predictions and errors of the models can be comprehensively visualized. Further, our study shows that pretrained sentence embeddings outperform even state-of-the-art contextual word embeddings and can be recommended as a suitable representation for this task. The full Python code is publicly available [6].

As persons spreading hate speech might be banned, penalized, or monitored not to put their threats into actions, prediction uncertainty is an important component of decision making and can help humans observers avoid false positives and false negatives. Visualization of prediction uncertainty can provide better understanding of the textual

[6] https://github.com/KristianMiok/Hate-Speech-Prediction-Uncertainty

context within which the hate speech appear. Plotting the tweets that are incorrectly classified and inspecting them can identify the words that trigger wrong classifications.

Prediction uncertainty estimation is rarely implemented for text classification and other NLP tasks, hence our future work will go in this direction. A recent emergence of cross-lingual embeddings possibly opens new opportunities to share data sets and models between languages. As evaluation in rare languages is difficult, the assessment of predictive reliability for such problems might be an auxiliary evaluation approach. In this context, we also plan to investigate convolutional neural networks with probabilistic interpretation.

Acknowledgments. The work was partially supported by the Slovenian Research Agency (ARRS) core research programme P6-0411. This project has also received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825153 (EMBEDDIA).

References

1. Baldi, P., Sadowski, P.J.: Understanding dropout. In: Advances in neural information processing systems. pp. 2814–2822 (2013)
2. Berger, W., Piringer, H., Filzmoser, P., Gröller, E.: Uncertainty-aware exploration of continuous parameter spaces using multivariate prediction. In: Computer Graphics Forum. pp. 911–920 (2011)
3. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. Journal of Machine Learning Research 13(Feb), 281–305 (2012)
4. Bleich, E.: The rise of hate speech and hate crime laws in liberal democracies. Journal of Ethnic and Migration Studies 37(6), 917–934 (2011)
5. Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., Niculae, V., Prettenhofer, P., Gramfort, A., Grobler, J., Layton, R., VanderPlas, J., Joly, A., Holt, B., Varoquaux, G.: API design for machine learning software: experiences from the scikit-learn project. In: ECML PKDD Workshop: Languages for Data Mining and Machine Learning. pp. 108–122 (2013)
6. Cer, D., Yang, Y., Kong, S.y., Hua, N., Limtiaco, N., John, R.S., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., et al.: Universal sentence encoder. arXiv preprint arXiv:1803.11175 (2018)
7. Chinchor, N.: Muc-4 evaluation metrics. In: Proc. of the Fourth Message Understanding Conference. p. 22–29 (1992)
8. Chollet, F., et al.: Keras. `https://keras.io` (2015)
9. Corazza, M., Menini, S., Arslan, P., Sprugnoli, R., Cabrio, E., Tonelli, S., Villata, S.: Comparing different supervised approaches to hate speech detection. In: EVALITA 2018 (2018)
10. Cox, J., Lindell, M.: Visualizing uncertainty in predicted hurricane tracks. International Journal for Uncertainty Quantification 3(2) (2013)
11. Davidson, T., Warmsley, D., Macy, M., Weber, I.: Automated hate speech detection and the problem of offensive language. In: Eleventh International AAAI Conference on Web and Social Media (2017)
12. Del Vigna12, F., Cimino23, A., Dell'Orletta, F., Petrocchi, M., Tesconi, M.: Hate me, hate me not: Hate speech detection on facebook (2017)
13. Fortunato, M., Blundell, C., Vinyals, O.: Bayesian recurrent neural networks. arXiv preprint arXiv:1704.02798 (2017)

14. Gal, Y., Ghahramani, Z.: Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In: International Conference on Machine Learning. pp. 1050–1059 (2016)
15. Gal, Y., Ghahramani, Z.: A theoretically grounded application of dropout in recurrent neural networks. In: Advances in Neural Information Processing Systems. pp. 1019–1027 (2016)
16. Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., Blei, D.M.: Automatic differentiation variational inference. The Journal of Machine Learning Research 18(1), 430–474 (2017)
17. Liu, L., Boone, A.P., Ruginski, I.T., Padilla, L., Hegarty, M., Creem-Regehr, S.H., Thompson, W.B., Yuksel, C., House, D.H.: Uncertainty visualization by representative sampling from prediction ensembles. IEEE transactions on visualization and computer graphics 23(9), 2165–2178 (2016)
18. Liu, L., Padilla, L., Creem-Regehr, S.H., House, D.H.: Visualizing uncertain tropical cyclone predictions using representative samples from ensembles of forecast tracks. IEEE Transactions on Visualization and Computer Graphics 25(1), 882–891 (2019)
19. McInnes, L., Healy, J., Saul, N., Grossberger, L.: UMAP: Uniform manifold approximation and projection. The Journal of Open Source Software 3(29), 861 (2018)
20. Mehdad, Y., Tetreault, J.: Do characters abuse more than words? In: Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue. pp. 299–303 (2016)
21. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
22. Miok, K.: Estimation of prediction intervals in neural network-based regression models. In: 20th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC). pp. 463–468 (09 2018)
23. Myshkov, P., Julier, S.: Posterior distribution analysis for bayesian inference in neural networks. In: Workshop on Bayesian Deep Learning, NIPS (2016)
24. Peters, M.E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., Zettlemoyer, L.: Deep contextualized word representations. arXiv preprint arXiv:1802.05365 (2018)
25. Pham, V., Bluche, T., Kermorvant, C., Louradour, J.: Dropout improves recurrent neural networks for handwriting recognition. In: 2014 14th International Conference on Frontiers in Handwriting Recognition. pp. 285–290. IEEE (2014)
26. Platt, J.C.: Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In: Advances in large margin classifiers. pp. 61–74. MIT Press (1999)
27. Rehurek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010)
28. Rother, K., Allee, M., Rettberg, A.: Ulmfit at germeval-2018: A deep neural language model for the classification of hate speech in german tweets. In: 14th Conference on Natural Language Processing KONVENS 2018. p. 113 (2018)
29. Ruginski, I.T., Boone, A.P., Padilla, L.M., Liu, L., Heydari, N., Kramer, H.S., Hegarty, M., Thompson, W.B., House, D.H., Creem-Regehr, S.H.: Non-expert interpretations of hurricane forecast uncertainty visualizations. Spatial Cognition & Computation 16(2), 154–172 (2016)
30. Schmidt, A., Wiegand, M.: A survey on hate speech detection using natural language processing. In: Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media. pp. 1–10 (2017)
31. Sparck Jones, K.: A statistical interpretation of term specificity and its application in retrieval. Journal of documentation 28(1), 11–21 (1972)
32. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research 15(1), 1929–1958 (2014)
33. Waldron, J.: The harm in hate speech. Harvard University Press (2012)

34. Wang, S., Manning, C.: Fast dropout training. In: International Conference on Machine Learning. pp. 118–126 (2013)
35. Warner, W., Hirschberg, J.: Detecting hate speech on the world wide web. In: Proceedings of the Second Workshop on Language in Social Media. pp. 19–26. Association for Computational Linguistics (2012)
36. Zaremba, W., Sutskever, I., Vinyals, O.: Recurrent neural network regularization. arXiv preprint arXiv:1409.2329 (2014)
37. Zhu, L., Laptev, N.: Deep and confident prediction for time series at uber. In: 2017 IEEE International Conference on Data Mining Workshops (ICDMW). pp. 103–110. IEEE (2017)