Multi-language Sentiment Classification Using Deep Convolutional Neural Networks

*** *** *** *** *** ***
****

ABSTRACT

This paper presents a novel approach to multi-lingual sentiment classification of short texts. In contrast to previously proposed multi-lingual approaches that typically require supervised data to establish a correspondence to English, our method does not require such supervision. We leverage large amounts of training data in various languages to train a multi-layer convolutional net and demonstrate the importance of using pre-training for such networks. We thoroughly evaluate our approach on various multi-lingual datasets, including recent SemEval-2016 sentiment prediction benchmark (Task 4) where our approach demonstrates state-of-the-art performance.

Keywords

Sentiment classification; multi-language; neural networks

1. INTRODUCTION

Automatic sentiment analysis is a fundamental problem in natural language processing (NLP). A huge volume of opinionated text is currently available on social media. On Twitter alone, 500 million tweets are published every day. This clearly highlights the need for automatically understanding the polarity and meaning of such texts. Sentiment analysis is a challenging task, due to the complexity of human language, where the use of rhetorical constructions such as sarcasm and irony easily confuse sentiment classifiers. Contextualization and informal language, which are often adopted in social media text, are additional complicating factors. Furthermore, the Internet is multi-lingual, and each language has its own grammar and syntactic rules. In 2016, only 26.3% of the total number of internet users are English speakers [14].

Considering the aforementioned difficulties, it is not surprising that the performance of existing commercial systems is still rather poor, as shown in several recent studies [8, 26]. The benchmark work of [26] showed that even the performance of the best systems largely varies across datasets and overall leaves much room for improvement. This clearly shows the importance of designing a method that generalizes well to different domains and languages.

Contributions. In this paper, we develop a multi-language sentiment classifier using minimal supervision. Our model is a multi-layer convolutional neural network (CNN) that learns to extract features that generalize to other languages. We exploit distant supervision to leverage large amounts of available weakly supervised data and show that it can be very effective to compensate the lack of labelled training examples, especially in a multi-language setting. We present extensive experiments quantifying the impact of the amount of weak labels used on the final prediction accuracy. Also, we investigate the interplay of pre-trained word-embeddings with the distant supervised and supervised training phases. All our experiments were performed on datasets in four languages, namely Dutch, English, German and Italian.

Our main contributions are summarized as follows:

- Our proposed simple CNN architecture substantially improves state-of-the-art results on text polarity classification in several languages. E.g., for English, our method outperforms all others on the SemEval-2016 Task 4 dataset [23], which is the most common benchmark for sentiment classification in English.
- We show that a single CNN model can be successfully trained for the joined task on all languages, as opposed to separate networks for each individual language. The joint approach has several advantages, such as removing the reliance on (possibly inaccurate) language identification systems as well as easier extensibility to new languages. We provide detailed comparison to similar per-language models, and show that the proposed joint model performs comparatively well. We even found some evidence of improved performance for the case of mixed-language tweets.
- We provide extensive practical guidelines on how to

WOODSTOCK '97 El Paso, Texas USA

DOI: 10.475/123_4

build and train CNN-based neural networks models for similar tasks.

2. RELATED WORK

In the following, we provide an overview of the most relevant works, related to the application of neural networks to sentiment classification, distant supervision and training multi-lingual text classifiers.

Neural networks. Neural networks have shown great promise in NLP over the past few years. Examples are semantic analysis [32], machine translation [11] and sentiment analysis [33]. In particular, shallow convolutional neural networks (CNNs) have recently improved the state-of-the-art in text polarity classification demonstrating a significant increase in terms of accuracy compared to previous state-of-the-art techniques [17, 16, 31, 15, 27, 2]. The most successful models for sentiment analysis use convolution neural networks, where a set of convolution filters acts as a sliding window over the input word sequence, typically followed by a pooling operation (such as max-pooling) to generate a fixed-vector representation of the input sentence.

CNNs vs RNNs. Recently, recurrent neural network architectures (RNNs), such as LSTMs, have received a lot of attention for various NLP tasks. Yet they have so far not managed to outperform convolutional architectures on polarity prediction tasks [28, Table 4]. This is evidenced by the recent SemEval-2016 challenge [23], where systems relying on convolutional networks rank at the top. In fact, long-term relationships captured well by LSTMs are of minor importance to the sentiment analysis of short tweets, whereas learning powerful n-gram feature extractors (which convolutional networks handle very well) that are able to effectively detect sentiment cues contributes much more to the discriminative power of the model. Additionally, LSTMs are much more computationally expensive than convolutional networks which still prevents their application to very large collections like the one used in this paper (300M tweets).

Distant-supervised learning. The use of semi-supervised or unsupervised learning has been an active research direction in machine learning and in particular NLP applications. Unsupervised training has been empirically shown to be beneficial for supervised machine learning tasks [10]. Distant pre-training is a variant of unsupervised learning that consists in inferring weak labels from data without manual labels. This approach has been used for text polarity classification where significantly larger training sets were generated from texts containing emoticons [12, 31]. [31] have shown that training a CNN on these larger datasets, followed by additional supervised training on a smaller set of manually annotated labels, yields improved performance for tweets.

Multi-language sentiment classification. Sentiment classification has drawn a lot of attention in the past few years both in industry and academia [23, 8]. Yet most of the research effort has been focusing on tweets written in one language (mostly English). One exception is the work of [6] that showed the existence of significant disparities between French, Dutch and English. Because English uses simpler vocabulary and syntactic constructions compared to French and Dutch, this can severely hinder the performance of a classifier trained on hand-crafted features.

The major factor that limits the development of accurate models for multi-lingual sentiment analysis is the lack of supervised corpora [4, 9]. Most of the existing approaches addressing this problem [20, 1] try to transfer knowledge from English – for which tools, labelled data and resources are abundant – to other languages for which resources are rather limited. An example is the approach introduced in [20], which transfers hand-crafted subjectivity annotation resources such as a per-word sentiment lexicon from English to Romanian. A similar approach introduced in [1] consists in translating a target language to English and then to use an English sentiment classifier rather than one specific to the target language. Several approaches have also been proposed to build distributed representations of words in multiple languages. The work of [35] used a Wikipedia corpus of five languages to train word embeddings, and then used anchor terms (names, cross-lingual words) to align the embeddings. [13] proposed a method to create bilingual word vectors by requiring words that are related across the two languages.

All the aforementioned approaches have the strong requirement of needing access to a set of correspondences between English and a target language. Some of these methods also require translating the target language to English. Yet machine translation is a very challenging task in NLP and represents an additional source of error in the classification system, due to various problems such as sparseness and noise in the data [9]. Additionally, such methods must crucially rely on an accurate language identification system, which is however a very difficult task, especially on short texts. See e.g. [19, 18] for an overview of such methods and their limitations in generalizing to different domains.

In this work, we present an approach to train a single model for all languages, which does not require to identify the language of input texts nor to translate it into a reference language. In a similar spirit, a Naïve Bayes classifier has been investigated in [24]. However, it relies on simple hand-crafted word-level features and hence does not attain a competitive performance. Our approach relies on a much more powerful model, i.e. a CNN architecture to automatically learn features from a large corpus, which significantly improves the-state-of-the-art performance on popular sentiment analysis benchmarks.

3. MODEL

Our model follows a multi-layer CNN architecture. Given an input sequence of words, the corresponding sequence of word-embeddings is feed as the input to the first convolutional layer. (each convolutional filter operating in a sliding window fashion over the text. Full details are described below). This layer is followed by max-pooling over the vari-

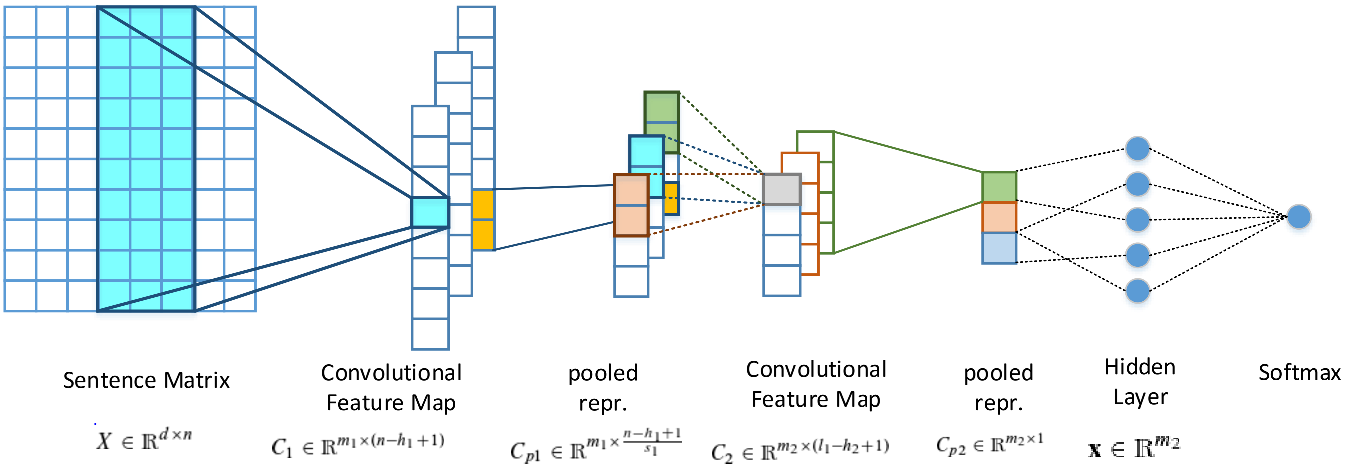

Sentence Matrix
$X \in \mathbb{R}^{d \times n}$
Convolutional Feature Map
$C_1 \in \mathbb{R}^{m_1 \times (n-h_1+1)}$
pooled repr.
$C_{p1} \in \mathbb{R}^{m_1 \times \frac{n-h_1+1}{s_1}}$
Convolutional Feature Map
$C_2 \in \mathbb{R}^{m_2 \times (l_1-h_2+1)}$
pooled repr.
$C_{p2} \in \mathbb{R}^{m_2 \times 1}$
Hidden Layer
$\mathbf{x} \in \mathbb{R}^{m_2}$
Softmax

Figure 1: The architecture of our CNN model with 2 convolutional layers

able length output, which is then feed into the next convolutional layer. Our architecture extends the single-layer CNN proposed by [31, 17, 16] to two and more convolutional layers. The resulting network architecture is illustrated in Figure 1 and in its basic variant consists in two consecutive pairs of convolutional-pooling layers followed by a single hidden layer and a soft-max output layer. In the following, we describe in detail each layer and corresponding parameters.

3.1 Convolutional Neural Network

Embedding layer. Each word is associated with a d-dimensional vector (embedding). An input sequence of n words is represented by concatenating their embeddings, yielding a sentence matrix $\mathbf{X} \in \mathbb{R}^{\mathrm{d}\times\mathrm{n}}$. $\mathbf{X}$ is used as input to the network.

Convolutional layer. This layer applies a set of m convolutional filters of length h over the matrix X. Let $\mathbf{X}_{[i:i+h]}$ denote the concatenation of word vectors $\mathbf{x}_i$ to $\mathbf{x}_{i+h}$. A feature c_i is generated for a given filter $\mathbf{F}$ by:

$$c_i := \sum_{k,j} (\mathbf{X}_{[i:i+h]})_{k,j} \cdot \mathbf{F}_{k,j} \tag{1}$$

The concatenation of all vectors in a sentence defines a feature vector $\mathbf{c} \in \mathbb{R}^{\mathrm{n-h+1}}$. The vectors $\mathbf{c}$ are then aggregated from all m filters into a feature map matrix $\mathbf{C} \in \mathbb{R}^{\mathrm{m}\times(\mathrm{n-h+1})}$. The filters are learned during the training phase of the neural network, as described in Section 3.2. The output of the convolutional layer is passed through a non-linear activation function, before entering a pooling layer.

Pooling layer. The pooling layer aggregates the input vectors by taking the maximum over a fixed set of non-overlapping intervals. The resulting pooled feature map matrix has the form: $\mathbf{C}_{\mathbf{pooled}} \in \mathbb{R}^{m \times \frac{n-h+1}{s}}$, where s is the length of each interval. In the case of overlapping intervals with a stride value s_t, the pooled feature map matrix has the form $\mathbf{C}_{\mathbf{pooled}} \in \mathbb{R}^{m \times \frac{n-h+1-s}{s_t}}$. Depending on whether the boundaries are included or not, the result of the fraction is rounded up or down respectively.

Hidden layer. A fully connected hidden layer computes the transformation $\alpha(\mathbf{W} * \mathbf{x} + \mathbf{b})$, where $\mathbf{W} \in \mathbb{R}^{\mathrm{m}\times\mathrm{m}}$ is the weight matrix, $\mathbf{b} \in \mathbb{R}^{\mathrm{m}}$ the bias, and α the rectified linear (`relu`) activation function [22]. The output $\mathbf{x} \in \mathbb{R}^{\mathrm{m}}$ of this layer can be seen as an embedding of the input sentence.

Softmax. Finally, the outputs of the previous layer $\mathbf{x} \in \mathbb{R}^{\mathrm{m}}$ are fully connected to a soft-max regression layer, which returns the class $\hat{y} \in [1, K]$ with largest probability, i.e.,

$$\begin{aligned} \hat{y} &= \arg\max_j P(y = j \mid \mathbf{x}, \mathbf{w}, \mathbf{a}) \\ &= \arg\max_j \frac{e^{\mathbf{x}^\intercal \mathbf{w}_j + a_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\intercal \mathbf{w}_k + a_j}}, \end{aligned} \tag{2}$$

where $\mathbf{w}_j$ denotes the weights vector of class j, from which the dot product with the input is formed, and a_j the bias of class j.

Network Parameters. The following parameters of the neural network are learned during training: $\theta = \{\mathbf{X}, \mathbf{F}_1, \mathbf{b}_1, \mathbf{F}_2, \mathbf{b}_2, \mathbf{W}, \mathbf{a}\}$, with $\mathbf{X}$ the word embedding matrix, where each row contains the d-dimensional embedding vector for a specific word; $\mathbf{F}_i, \mathbf{b}_i$ the filter weights and biases of convolutional layers; and $\mathbf{W}$ and $\mathbf{a}$ the weight-matrix for output classes in the soft-max layer.

3.2 Learning the Model Parameters

The model parameters are learned using the following three-phase procedure: (i) creation of word embeddings; (ii) distant-supervised phase, where the network parameters are tuned by training on weakly labelled examples; and (iii) final supervised phase, where the network is trained on the supervised training data.

Preprocessing and Word Embeddings. The word embeddings are initialized using word2vec [21] and further tuned on an unsupervised corpus of $300M$ tweets. We apply a skip-gram model of window-size 5 and filter words that occur less than 15 times [31]. The dimensionality of the vector representation is set to $d = 52$.

Training. During the first distant-supervised phase, we

Table 1: Data used for unsupervised pre-training of word embeddings and distance supervision. Note that the Dutch datasets does not contain any neutral labels.

Language	Dataset	Total	Neutral	Neg.	Pos.
Dutch	Pre-training	40M	-	11M	29M
	Training		-	513	532
	Test		-	65	51
English	Pre-training	60M	-	30M	30M
	Training		7544	2748	7752
	Validation		987	365	1038
	Test		10342	3231	7059
German	Pre-training	40M	-	8M	32M
	Training		5319	1443	2193
	Test		567	177	250
Italian	Pre-training	40M	-	10M	30M
	Training		2942	2293	1434
	Test		314	250	177

use emoticons to infer noisy labels on the tweets in the training set [25, 12]. The details of the training set obtained from this procedure are listed in Table 1, "Pre-training" rows. The neural network was trained on these data for one epoch, before finally training on the supervised data for about 20 epochs. The word-embeddings $\mathbf{X} \in \mathbb{R}^{d \times n}$ are updated during both the distant- and the supervised training phases by applying back-propagation through the entire network. The resulting effect on the embeddings of words are illustrated in Figure 4 and discussed in Section 4.4.

Optimization. During both training phases, the network parameters are learned using *AdaDelta* [36]. We compute the score on the validation set at fixed intervals and select the parameters achieving the highest score.

4. EXPERIMENTAL RESULTS

4.1 Data

We used a large unsupervised corpus of approximately $300M$ tweets written in four different languages, namely Dutch, English, German, and Italian. Our collected tweets each containing at least one emoticon (positive or negative smiley). The smileys are used to automatically infer weak labels (and subsequently removed from the tweets). This idea of distant-supervised technique was originally described in [12, 31, 30]. We used the same set of tweets to construct the word embeddings (unsupervised). For the final supervised phase as well as evaluation, we used available public labeled benchmark datasets. While the manually labeled English [23], Italian [29] and Dutch [34] corpora are publicly available, the German dataset [3] will be made available by the authors. An overview of all used datasets as well as the number of labelled tweets is given in Table 1.

Data Preparation. Each tweet was preprocessed in three steps: (i) URLs and usernames were substituted by a replacement token, (ii) the text was lowercased and (iii) finally tokenized using the NLTK tokenizer.

4.2 Baselines

To better understand the importance of the multi-language aspect of our approach, we compared different approaches to train our neural network:

- A single-language approach *SL-CNN*, where we train a separate model for each of the four languages in the corpus. We used a set of $300M$ tweets per language for the pre-training phase

- A multi-language model *ML-CNN*, where the distant-supervised phase was performed jointly for all languages at once, and the final supervised phase independently for each language. For the pre-training phase, we used a balanced set of $300M$ tweets that included all four languages, see Table 1, "Pre-training".

- A fully multi-language model *FML-CNN* where all training phases were performed without differentiation between languages. The pre-training set is the same as in *ML-CNN*

We also compared the results of our approach to a common baseline found in literature, i.e. a random forest (RF) classifier trained on n-gram features, as proposed in [24]. In addition, results on the English datasets were compared to the best known ones from the SemEval benchmark[1].

4.3 Evaluation strategy

We evaluate the performance of the proposed models using the average macro F1-score of the positive and negative classes. Each approach was trained for a fixed number of epochs and then we selected the results which yielded the best results on a separate validation set.

For *SL-CNN* and *ML-CNN* the validation set consists of a single language set. For English we used the `test2015` set as validation set and the `test2016` for testing from SemEval-2016 challenge. For *FML-CNN* the validation is the combination of the validation sets used for each language.

4.4 Results

The F1-scores of the proposed approaches on all four tested languages are summarized in Table 2. Overall, the scores for Dutch are much higher than the other three languages as the Dutch dataset does not contain any neutral class. When comparing three CNN variants as well as the RF baseline, we see that *SL-CNN* gets slightly better scores than *ML-CNN* or *FML-CNN*. The difference in terms of performance is rather small, i.e. $+1.3\%$ on average for English, German and Italian while *ML-CNN* performs better for Dutch. We believe this is a compelling argument for using the simpler*ML-CNN* system, which does not need language identification (which is challenging for short texts). As we will discuss below, *ML-CNN* in particular performs significantly better on mixed language texts.

[1] http://alt.qcri.org/semeval2016/

Table 2: *Comparison of the F1-scores of the three CNN variants as well as several baselines.* We also compare to the top three systems from the SemEval 2016 competition [23]. The highest scores among the three models we propose are highlighted in bold face. For the English tweets, we additionally compare results to the best reported scores from the SemEval-2016 competition (bottom rows). English-val consists of the 2015 SemEval test set. Note that although *SL-CNN* performs slightly better on average, the difference with *ML-CNN* is rather small.

Method	**Test Set Language**				
	Dutch	**English-val**	**English-test**	**German**	**Italian**
SL-CNN	88.33	**66.33**	**63.58**	**64.19**	**65.87**
ML-CNN	**91.31**	64.76	61.61	63.62	64.73
FML-CNN	85.92	62.85	61.03	63.19	64.80
RF	81.03	49.62	48.60	52.40	52.71
SENSEI-LIF [23]	-	66.16	62.96	-	-
UNIMELB [23]	-	65.05	61.67	-	-

***SL-CNN* vs *ML-CNN*.** One benefit of the multi-language models over the single-language ones is their ability to deal with text in mixed languages. In order to check this hypothesis, we used the *langpi* tool [19] to extract a set of 300 tweets from the German corpus that contain English words in them. Note that although these tweets were classified by Twitter as German, they nevertheless contain a significant number of English words (some of them being entirely written in English). We also manually inspected this set and discarded tweets that did not contain English. We then retrained the two models discarding the set of 300 tweets from the training set. When evaluating on this subset, the *ML-CNN* obtained an F1-score of 68.26 while *SL-CNN* obtained 64.07. When manually inspecting the results, it became clear that the *ML-CNN* was better at classifying tweets that were entirely in English or contained a significant number of English words.

Leveraging Distant Training Data. Figure 2 compares the F1-scores for each language when changing the amount of data used in the distant-supervised phase. For the multi-language approach, the sets used for the distant-supervised phase are balanced in terms of the numbers of tweets used per language. The scores without distant supervision are the lowest for all languages. We typically observe an increase in terms of F1-score when increasing the amount of training data. The performance gain for Dutch is substantial (around 10%) while it is more moderate – but still significant – for Italian (around 1%).

Word Embeddings. To investigate the importance of the initialization of word embeddings and their interaction with the distant supervised phase, we compared the system performance of ML-CNN and FML-CNN in four scenarios: (i) using randomly initialized word embeddings, not updated during the distant-supervised phase (named *Full Random CNN*), (ii) using randomly initialized word embeddings, updated during the distant-supervised phase (*Random Word Embeddings*), (iii) using word2vec embeddings without distant supervision (*No Distant Supervision*) and (iv) using word2vec embeddings, updated during the distant-supervised phase using $160M$ tweets (*Fully trained CNN*). Results shown in Figure 3 demonstrate that the *Fully trained CNN* approach performed the best in almost all cases, although we did observe that (iii) performed better for *ML-CNN* on the Italian corpus. These results prove that the quality of the initialization as well as updating the large number of word vector parameters during the training of the network yield significant improvements.

Figure 4 illustrates the effect of the distant-supervised phase on the word embeddings. For visualization purposes, principal component analysis (PCA) was used to project the word embeddings to two dimensions. We made two observations from this experiment. First, we notice that in the case of the multi-language models (*FML-CNN*), the distant-supervised training phase significantly changes the geometry of the word embeddings, creating a space where the words coming from different languages are further apart. Second, we see that the geometry of the word embeddings also reflects the distance in terms of sentiment between pairs of words (*SL-CNN*). Figure 4(b) shows the initial word embeddings created by word2vec, before the distant-supervised phase. Taking as an example the pair of words "good" and "bad", it is clear that these two words often appear in the same context and are thus close to each other in the embedded space. The similarity score of these two vectors is 0.785. After the distant-supervised phase, the semantic of the space is changed and the distance between words come to reflect the difference in terms of sentiment. As shown in Figure 4(c), negative and positive words are neatly separated into two clusters. In this case, the similarity score between the word embeddings of "good" and "bad" becomes -0.055. Finer grained clusters are also revealed in the second embedding. For example, words that convey sadness are close together.

Distant- and Supervised Phases. We investigated the effect of the distant-supervised phase on the final training and test scores, based on the datasets listed in Table 1. We trained each CNN for a maximum of 120 epochs and computed the F1-score at regular intervals of 5 times per epoch. We show the results on the test sets in Figure 5 for different variants of distant pre-training, for both multi-language architectures. For the test score, we also show the results

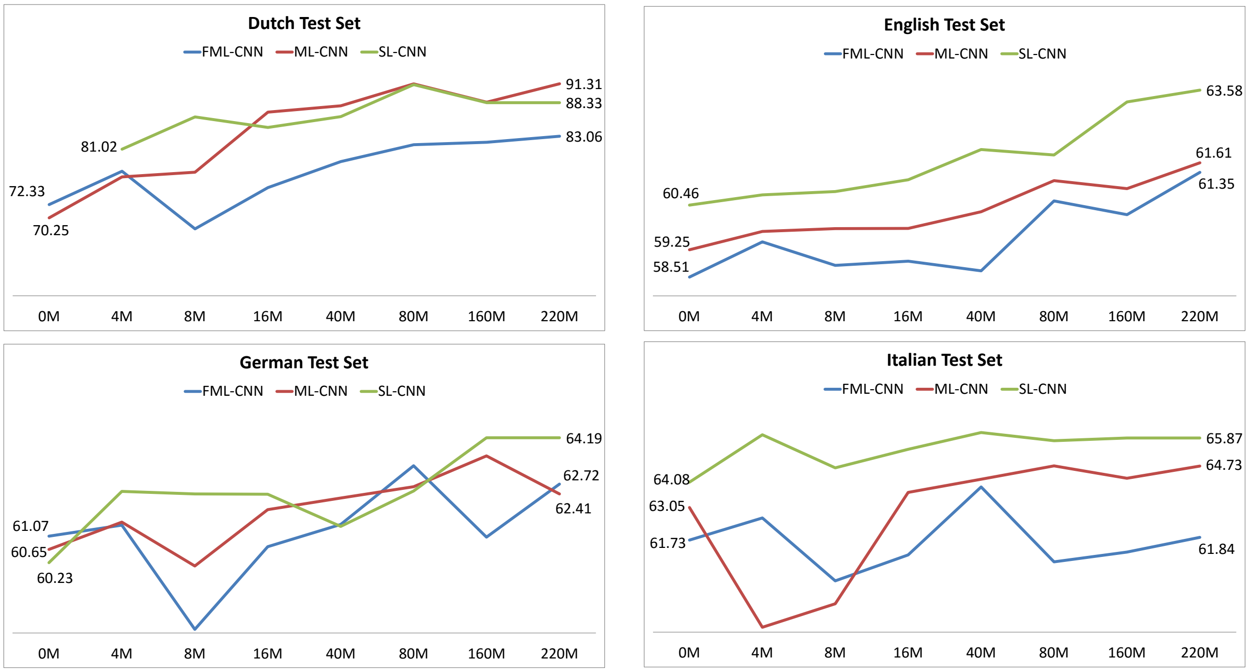


Figure 2: *Results obtained over varying the amount of data during the distant supervised phase.* Each CNN was trained for one epoch. There are two things to note: first, we rescaled the curve for *SL-CNN* to match the amount of data used per language by the multi-language approaches. So while the multi-language approaches were trained with, for example, $40M$ tweets (i.e. $10M$ tweets per language), each *SL-CNN* model was trained with $10M$ tweets, all coming from the same language. Second, while each set up through $160M$ contains the same number of tweets per language, the set containing $220M$ is unbalanced as it contains $100M$ tweets in English and the rest divided equally between Dutch, German and Italian.

without distant pre-training as well as for varying sizes of the training set used in the distant-supervised phase. The point-markers show where the maximum score values were obtained.

Note that the time it takes for a method to get to a regime of good predictive performance highly depends on the use of the distant-supervised pre-training. The networks trained with distant-supervised data achieve their best performance much earlier than the network trained without distant supervision.

Comparing Network Architectures. We investigated several network architectures for the task of interest. Figure 6 compares the F1-scores of our single language model *SL-CNN* as described above, varying the number of convolutional/pooling layer pairs from 1 to 3. We observe that as the number of layers and therefore network parameters increases, more and more training data is required in the distant-supervised phase for successful training, i.e, the capacity of the network improves with the number of layers. Therefore in practice and for new languages, we recommend to use this as a guideline depending on the amount of available training data. For the task of sentiment classification, current recurrent architectures such at LSTMs still perform inferior to CNNs currently, see e.g. [28, Table 4] and the discussion in the related work section.

4.5 Implementation Details

The core routines of our system are written in `Theano` [5] exploiting GPU acceleration with the `CuDNN` library [7]. The whole learning procedure takes approximately 24-48 hours to create the word embeddings, 20 hours for the distant-supervised phase with $160M$ tweets and only 30 minutes for the supervised phase with $35K$ tweets.

Experiments were conducted on 'g2.2xlarge' instances of *Amazon Web Services* (AWS) with *GRID K520* GPU having 3072 CUDA cores and 8 GB of RAM. The source code will be made available upon publication.

5. DISCUSSION

Needs more work...

Conclusion. We described a deep learning framework to predict the sentiment polarity of short texts, such as tweets, written in multiple languages. Through a thorough experimental evaluation, we addressed some fundamental questions around the performance of such models. First, we demonstrated that the strategy used to train such models plays an important role in the obtained performance. Two important factors are a good initialization for the word vectors as well as pretraining using large amounts of weakly supervised data. Second, we contrasted the performance of a single-language and multi-language approach. The single-language model reaches the best performance and it even outperforms existing state-of-the-art methods on all the datasets of the

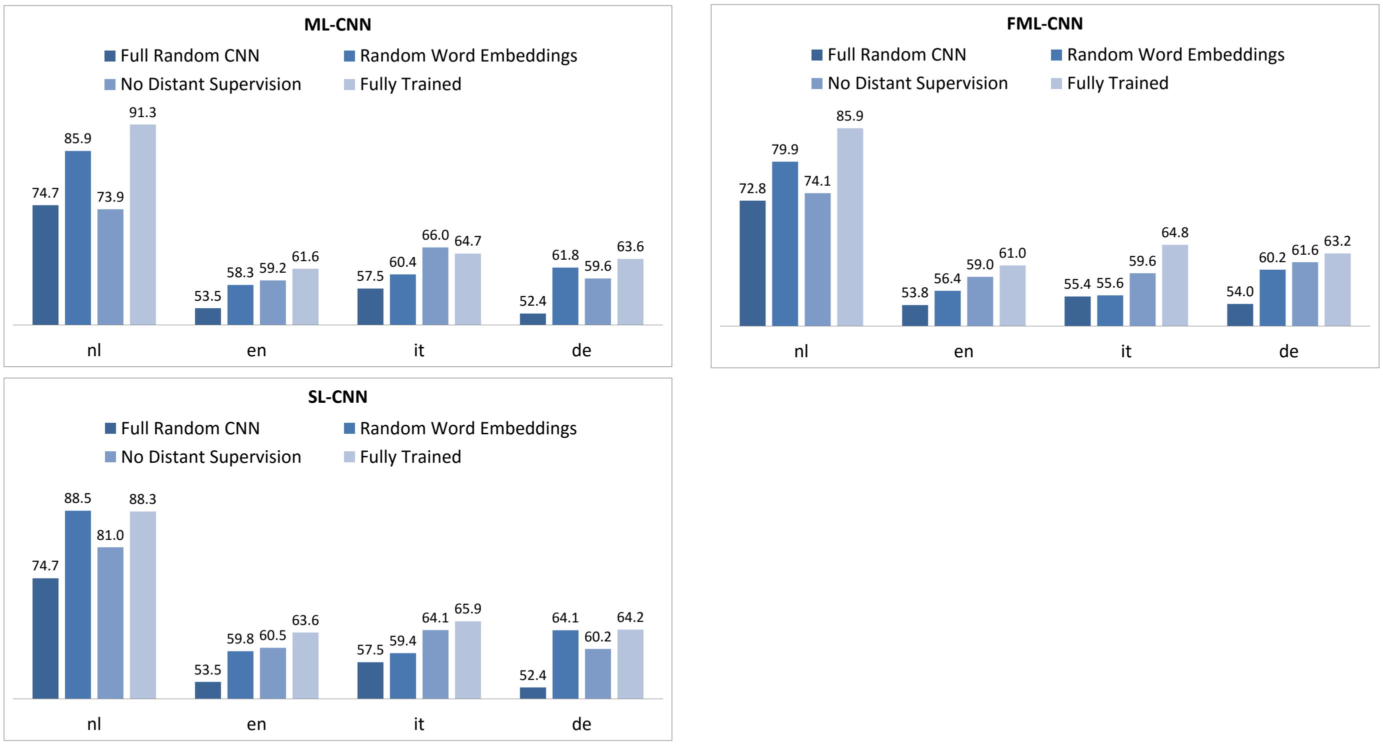
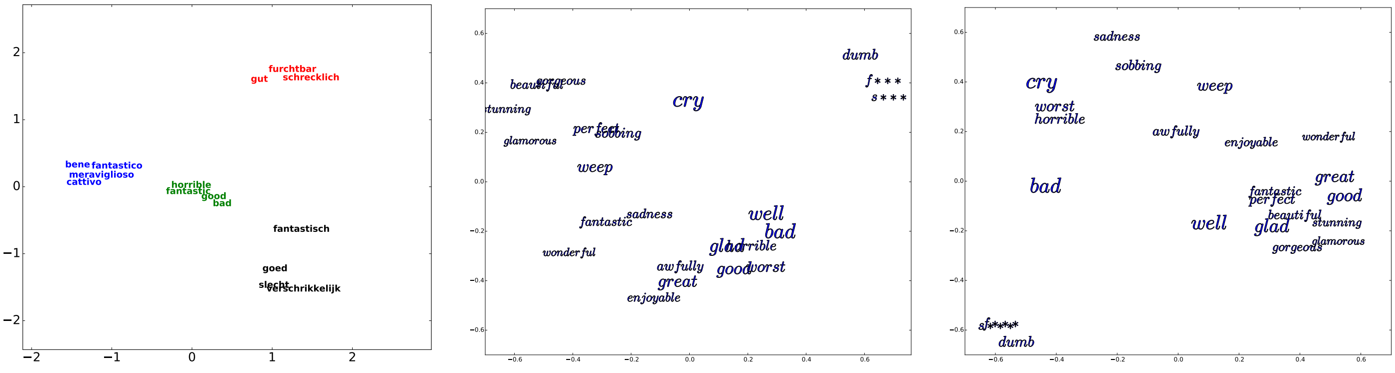


Figure 3: *Results obtained by initializing the CNNs with different word embeddings.* The fully trained variant typically performs better than the other three variants thus demonstrating the importance of initializing the words vectors as well as performing distant-supervised training.



(a) (b) (c)

Figure 4: *Word embeddings projected onto a 2-dimensional space using PCA after the distant-supervised training phase.* (a) Words embeddings from the *ML-CNN* model. We used one color per language. We observe that the training phase significantly changes word embeddings towards a geometry that better reflects the disparity between languages. (b)-(c) Word embeddings before and after the distant-supervised phase. We see how the words vectors move to reflect disparities in terms of sentiments.

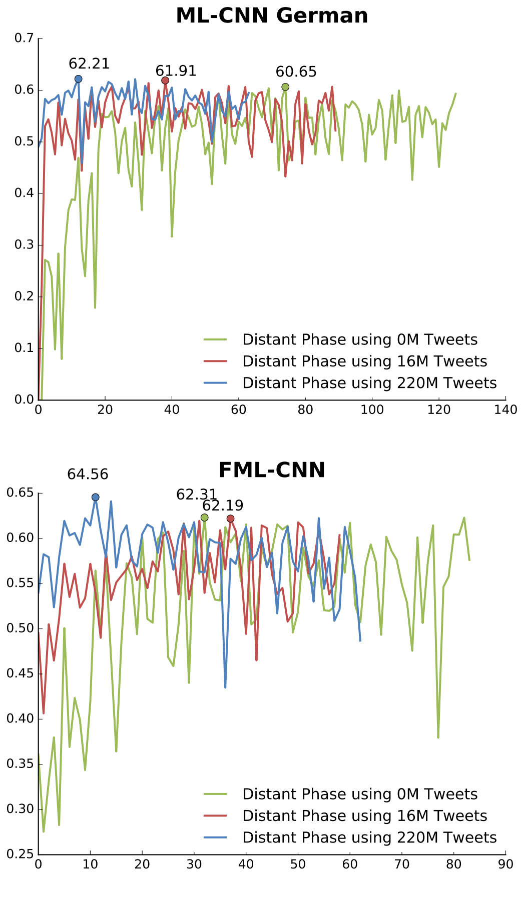


Figure 5: *Test F1-score of ML-CNN (on the German dataset) and FML-CNN vs. number of epochs during the final supervised phase.* We also show the effect of using a different amount of tweets in the distant supervised phase. Point-markers indicate the location of the maximum score.

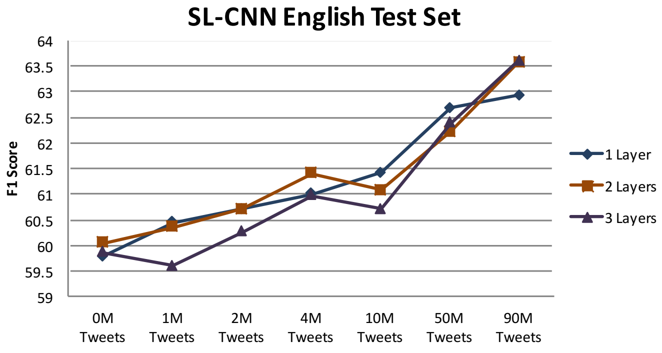


Figure 6: Results obtained by training the *SL-CNN* models with different number of layers, depending on different amounts of data during the distant-supervised phase. Each CNN was trained for one epoch.

SemEval-2016 competition. The multi-language approach performs slightly worse than its single-language counterpart but it is nevertheless competitive and exhibits several advantages: no need to know *a priori* the language (or languages) used in each tweet; the model model can be easy extended to more languages; it can cope with texts written in multiple languages.

6. REFERENCES

[1] M. Araujo, J. Reis, A. Pereira, and F. Benevenuto. An evaluation of machine translation for multilingual sentence-level sentiment analysis. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 1140–1145. ACM, 2016.

[2] author citation. anonymous. 2016.

[3] author citation. anonymous. 2016.

[4] A. Balahur and M. Turchi. Multilingual sentiment analysis using machine translation? In *Proceedings of the 3rd Workshop in Computational Approaches to Subjectivity and Sentiment Analysis*, 2012.

[5] J. Bergstra, O. Breuleux, F. F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: a CPU and GPU math compiler in Python. *Proceedings of the Python for Scientific Computing Conference (SciPy)*, pages 1–7, 2010.

[6] E. Boiy and M.-F. Moens. A machine learning approach to sentiment analysis in multilingual web texts. *Information retrieval*, 12(5):526–558, 2009.

[7] S. Chetlur and C. Woolley. cuDNN: Efficient Primitives for Deep Learning. *arXiv preprint arXiv: . . .* , pages 1–9, 2014.

[8] M. Cieliebak, O. Dürr, and F. Uzdilli. Potential and limitations of commercial sentiment detection tools. In *ESSEM@ AI* IA*, pages 47–58. Citeseer, 2013.

[9] K. Dashtipour, S. Poria, A. Hussain, E. Cambria, A. Y. A. Hawalah, A. Gelbukh, and Q. Zhou. Multilingual sentiment analysis: State of the art and independent comparison of techniques. *Cognitive Computation*, 2016.

[10] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11:625–660, 2010.

[11] J. Gao, X. He, W.-t. Yih, and L. Deng. Learning continuous phrase representations for translation modeling. In *ACL*, 2014.

[12] A. Go, R. Bhayani, and L. Huang. Twitter Sentiment Classification using Distant Supervision. Technical report, The Stanford Natural Language Processing Group, 2009.

[13] S. Gouws, Y. Bengio, and G. Corrado. Bilbowa: Fast bilingual distributed representations without word alignments. In *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[14] IWS. http://www.internetworldstats.com/stats7.htm, 2016.

[15] R. Johnson and T. Zhang. Semi-supervised Convolutional Neural Networks for Text Categorization via Region Embedding. In *NIPS 2015 - Advances in Neural Information Processing Systems 28*, pages 919–927, 2015.

[16] N. Kalchbrenner, E. Grefenstette, and P. Blunsom. A Convolutional Neural Network for Modelling Sentences. In *ACL - Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 655–665, Baltimore, Maryland, USA, Apr. 2014.

[17] Y. Kim. Convolutional Neural Networks for Sentence Classification. In *EMNLP 2014 - Empirical Methods in Natural Language Processing*, pages 1746–1751, Aug. 2014.

[18] M. Lui and T. Baldwin. Cross-domain feature selection for language identification. In *In Proceedings of 5th International Joint Conference on Natural Language Processing*. Citeseer, 2011.

[19] M. Lui and T. Baldwin. langid. py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 system demonstrations*, pages 25–30. Association for Computational Linguistics, 2012.

[20] R. Mihalcea, C. Banea, and J. M. Wiebe. Learning multilingual subjective language via cross-lingual projections. In *ACL*, 2007.

[21] T. Mikolov, Q. V. Le, and I. Sutskever. Exploiting Similarities among Languages for Machine Translation. *arXiv*, Sept. 2013.

[22] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 807–814, 2010.

[23] P. Nakov, A. Ritter, S. Rosenthal, V. Stoyanov, and F. Sebastiani. SemEval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation*, SemEval '16, San Diego, California, June 2016. Association for Computational Linguistics.

[24] S. Narr, M. Hulfenhaus, and S. Albayrak. Language-independent twitter sentiment analysis. *Knowledge Discovery and Machine Learning (KDML), LWA*, pages 12–14, 2012.

[25] J. Read. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of the ACL student research workshop*, pages 43–48. Association for Computational Linguistics, 2005.

[26] F. N. Ribeiro, M. Araújo, P. Gonçalves, F. Benevenuto, and M. A. Gonçalves. A benchmark comparison of state-of-the-practice sentiment analysis methods. *arXiv preprint arXiv:1512.01818*, 2015.

[27] S. Rothe, S. Ebert, and H. Schutze. Ultradense Word Embeddings by Orthogonal Transformation. *arXiv*, Feb. 2016.

[28] S. Semeniuta, A. Severyn, and E. Barth. Recurrent Dropout without Memory Loss. *arXiv*, 2016.

[29] sentipolc. http://www.di.unito.it/~tutreeb/sentipolc-evalita16/data.html, 2016.

[30] A. Severyn and A. Moschitti. Twitter Sentiment Analysis with Deep Convolutional Neural Networks. In *38th International ACM SIGIR Conference*, pages 959–962, New York, New York, USA, 2015. ACM Press.

[31] A. Severyn and A. Moschitti. UNITN: Training Deep Convolutional Neural Network for Twitter Sentiment Classification. In *SemEval 2015 - Proceedings of the 9th International Workshop on Semantic Evaluation*, 2015.

[32] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 101–110. ACM, 2014.

[33] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

[34] B. Verhoeven and W. Daelemans. Clips stylometry investigation (csi) corpus: A dutch corpus for the detection of age, gender, personality, sentiment and deception in text. In *LREC*, pages 3081–3085, 2014.

[35] M. Wick, P. Kanani, and A. Pocock. Minimally-constrained multilingual embeddings via artificial code-switching. 2015.

[36] M. D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. *arXiv*, page 6, 2012.