A Character-based Convolutional Neural Network for Language-Agnostic Twitter Sentiment Analysis

Jônatas Wehrmann*, Willian Becker*, Henry E. L. Cagnini*, and Rodrigo C. Barros†
Faculdade de Informática
Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681, 90619-900, Porto Alegre, RS, Brazil
* Email: {jonatas.wehrmann,willian.becker,henry.cagnini}@acad.pucrs.br
† Email: rodrigo.barros@pucrs.br

***Abstract*—Most work on tweet sentiment analysis is monolingual and the models that are generated by machine learning strategies do not generalize across multiple languages. Cross-language sentiment analysis is usually performed through machine translation approaches that translate a given source language into the target language of choice. Machine translation is expensive and the results that are provided by theses strategies are limited by the quality of the translation that is performed. In this paper, we propose a language-agnostic translation-free method for Twitter sentiment analysis, which makes use of deep convolutional neural networks with character-level embeddings for pointing to the proper polarity of tweets that may be written in distinct (or multiple) languages. The proposed method is more accurate than several other deep neural architectures while requiring substantially less learnable parameters. The resulting model is capable of learning latent features from all languages that are employed during the training process in a straightforward fashion and it does not require any translation process to be performed whatsoever. We empirically evaluate the efficiency and effectiveness of the proposed approach in tweet corpora based on tweets from four different languages, showing that our approach comfortably outperforms the baselines. Moreover, we visualize the knowledge that is learned by our method to qualitatively validate its effectiveness for tweet sentiment classification.**

Keywords*—Sentiment Analysis, Twitter, Convolutional Neural Networks, Deep Learning, Character-based Embedding.

I. Introduction

In the social networks era, people often share their own opinions about a variety of topics on the Internet. Twitter is one of the largest microblogging services avalilable with a substantial amount of data being generated every day. Its content constitutes a rich source for researchers in different scientific fields such as machine learning (ML) and natural language processing (NLP). Many efforts have been dedicated to accurately perform sentiment analysis for Twitter corpora over the past few years [1]. In a nutshell, sentiment analysis on Twitter data consists on assigning a polarity to each tweet, which can be positive, neutral, or negative regarding its main discussed topic. Tweets are short 140-character text that represent the opinions of a given user regarding the discussed topic, often reflecting the emotional state of mind of their authors with respect to the subject being approached.

Tweet sentiment analysis is usually language-centric, which means that the models derived from the learning process are mostly not applicable across distinct languages. Therefore, if one could build a classification approach that is not restricted to analyzing a single language, much more data regarding a given subject could be collected, eventually leading to more robust sentiment classification models. The most common approach for multilingual sentiment analysis is called Cross-Language Sentiment Classification (CLSC), which mainly focuses on the use of machine translation techniques for translating a given source language to the desired target language [2]–[6].

There are typically two scenarios in which machine translation is used in the CLSC context: (i) to translate training data and then perform learning and classification in the target language; and (ii) to translate test data and then perform learning and classification in the source language [7]. Notwithstanding, due to the intrinsic differences between languages, several problems may occur after translation. One of them is when a word that frequently appears in the source language may rarely appear in the target language after translation, generating a severe discrepancy in the data distribution between source and target languages [8]. Even if one could have perfect translation for a particular corpus, there is also the potential cultural distance between source and target languages, that may largely influence the final classification performance [9]. Another disadvantage of machine translation approaches concerns the availability of state-of-the-art open-source translators. Indeed, most of the translation APIs are not free of charge, so the task of translating large corpora may end up being too costly.

Deep neural networks (DNNs) have recently achieved significant performance gains in a variety of NLP tasks such as language modeling [10], sentiment analysis [11], syntactic parsing [12], and machine translation [13]. Convolutional neural networks (CNNs) and Long short-term memory networks (LSTMs) have been extensively employed for sentiment classification [14]–[16]. However, automatic sentiment classification of (unstructured) text data requires documents to be modeled as structured data so they can be interpreted by the ML algorithm. One of the solutions for this problem is to represent words as dense vectors, which was firstly introduced in [10] for the context of neural language modeling, and it was first applied to NLP tasks in the pioneering work of Collobert et al. [12], [17]. Several distinct *embedding* strategies for converting free text into vectors have arisen recently, each with their advantages and disadvantages. Inspired by the success of recent work on monolingual feature representation, much effort has been dedicated towards learning suitable feature representations for DNNs in the context of cross-lingual sentiment analysis, specially in the case of learning bilingual representation [7],

[18]. However, no approach to date can jointly deal with four distinct languages in a translation-free fashion.

With that in mind, our contributions in this paper are as follows. First, we present an effective deep neural architecture for learning feature representations of four distinct languages at once. To the best of our knowledge, we are the first to present a language-agnostic deep neural approach that learns multiple different languages simultaneously. Second, the proposed architecture is not just effective but highly efficient, given that it comprises substantially less parameters to be learned while reaching results that outperform the current state-of-the-art such as LSTMs. Third, we explore the gradients of the generated model in order to build a visualization strategy that allows to visually detect the importance of sequences of characters in the final classification. Finally, we make our collected Twitter dataset with four labeled languages publicly-available for research and experimental reproducibility purposes.

The remainder of this paper is organized as follows. Section II presents an overview on related work. Section III describes the problem statement, i.e., the precise task that we are addressing in this work. Section IV presents strategies for learning text representations in deep neural networks, and in particular the proposed character-based architecture for language-agnostic classification. Section V details the experimental methodology that is used for performing the empirical analysis, described in turn in Section VI. In Section VII, we describe our approach for visualizing what was learned by the proposed method over sequences of characters. Finally, Section VIII concludes the paper and points to future research directions.

II. Related Work

Sentiment classification techniques can be roughly divided into three approaches: ML-based, lexicon-based, and hybrid [19]. ML approaches address sentiment analysis as a regular text classification problem that makes use of syntactic and/or linguistic features [20]. Whereas some approaches identify the aspects that are being discussed together with their polarity (e.g., hotel reviews) [21], others simply assign an overall polarity to the entire document (e.g., movie reviews) [22].

Cross-language sentiment classification is often based on machine translation strategies to reduce the data to a single language [4]. Such an approach translates training and test data into the same target language, and then allows the application of a monolingual classifier. However, classification performance is often negatively affected due to the cultural gap between source and translated languages [23]. Many studies explore bilingual sentiment classification [18], [24]–[27]. Mihalcea et al. [24] make use of an English corpora to train sentence-level subjectivity classifiers in Romanian language using two approaches. In the first approach, they use a bilingual dictionary to translate an existing English lexicon to build a target language subjectivity lexicon. In the second one, they generate a subjectivity-annotated corpus in a target language by projecting annotations from an automatically-annotated English corpus. The authors argue that both approaches can be applied to any language, and not only Romanian. In [25], the authors propose an attention-based LSTM network to learn the document representations of reviews in English and Chinese exploring word vectors as text representation. Zhou et al. [18] propose a bilingual sentiment word embedding algorithm based on denoising autoencoders by learning 2,000 sentiment words. A bilingual constraint is also explored in [26] to build a consistent embedding space across languages by learning pairs of translated sentences.

It is important to notice that, to the best of our knowledge, no work to date proposes a translation-free language-agnostic approach that is capable of dealing with multiple (e.g., > 2) languages at once, which is the case of the method proposed in this paper. We detail it in the next sections.

III. Problem Statement

In this paper, we are concerned with a particular task that differs from cross-language learning, multilingual embedding learning, and similar approaches. In this task, we assume the existence of a multilingual corpora $\mathcal{C}$, in which $\mathcal{C}_i \in \mathcal{C}$ is composed by a triple $(\mathcal{T}_i, \mathcal{L}_i, \mathcal{P}_i)$, where $\mathcal{T}_i$ is a given document/sentence/text, $\mathcal{L}_i$ is the main language of the document, and $\mathcal{P}_i$ is the document polarity. Thus, one must approximate a mapping function σ so that $\sigma(\mathcal{T}_i) = \hat{\mathcal{P}}_i$, where $\hat{\mathcal{P}}_i$ is the estimated polarity for the i^{th} document of the multilingual corpora. Note that such mapping must be performed regardless of the document's source language $\mathcal{L}_i$. In this task, we assume that each language corpus contains enough resources for learning function σ, eliminating the necessity of machine translation tools.

A language-agnostic sentiment analysis method must be capable of: (i) learning directly from the documents present in the multilingual unpaired corpora; (ii) predicting the text polarity regardless of the source language; (iii) providing predictions even for multilingual documents, i.e., documents that contain words from multiple languages at the same time. In this paper, the annotated corpora $\mathcal{C}$ comprises tweets $\mathcal{T}_i$, where $\mathcal{L}_i \in \{English, German, Portuguese, Spanish\}$ and $\mathcal{P}_i \in \{Negative, Positive\}$.

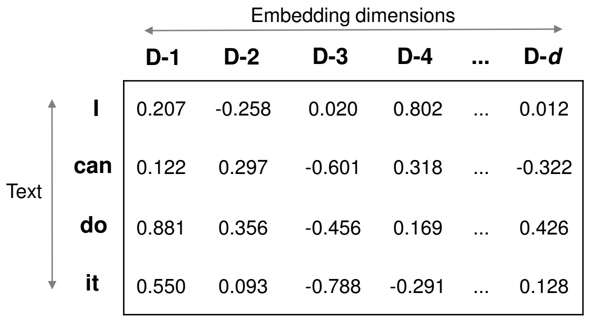
IV. Learning Text Representations

One key aspect of ML-based sentiment classification is regarding the feature space representation. One way of representing words is through dense vectors, where each word is embedded in a d-dimensional vector space [10], [28]. Instead of using words, one can also use characters, in a strategy similar to the one used in [29]–[31]. Character-based embedding was recently introduced in the context of convolutional neural networks [15], [32], and it is said to be better-suited for machine translation than its word-based counterparts [13].

A. Word-level Embedding

Word-level embedding consists in mapping word ω into a d-dimensional space in which semantically-similar words are neighbors. For instance, in single-language analysis, words such as $nice$ and $cool$ should be close to each other within the generated d-dimensional feature space. In the multilingual scenario, this property may be advantageous since similar words in different languages should lie close in the embedded d-dimensional space. However, word-level embedding requires as input a vocabulary with several words, posing two major shortcomings: (i) for multilingual analysis, the vocabulary size grows with the number of languages that are employed; and (ii) *false cognates* – words syntactically identical but with different meanings across languages – will most certainly confuse the ML algorithm and harm classification performance.

Let $\mathcal{T} \in \{\omega_1, \omega_2, ..., \omega_n\}$ be a text with n words and $\phi(\omega_i) = v_i$ be a function that maps word w_i into vector v_i, the text representation in a word-embedding space is defined by $\Gamma \in \mathbb{R}^{n \times d}$. Note that n varies with the size of text $\mathcal{T}$. Figure 1 depicts an example of the word-based representation.


Embedding dimensions
Text

	D-1	D-2	D-3	D-4	...	D-*d*
I	0.207	-0.258	0.020	0.802	...	0.012
can	0.122	0.297	-0.601	0.318	...	-0.322
do	0.881	0.356	-0.456	0.169	...	0.426
it	0.550	0.093	-0.788	-0.291	...	0.128

Fig. 1. Word-level representation of sentence "I can do it".

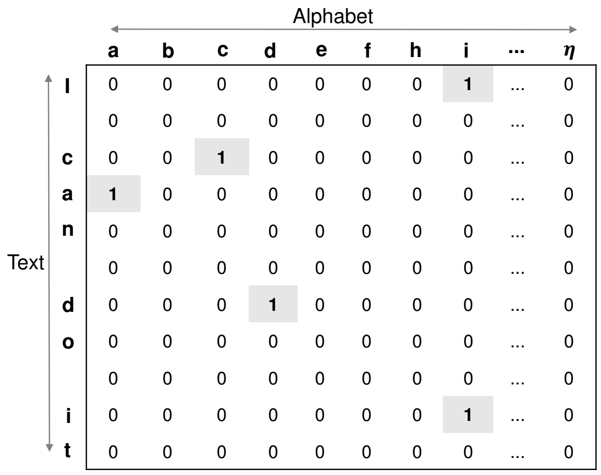
For training multilingual classifiers with word-level embedding, one needs to build a large dataset with instances from the desired languages. In the context of Twitter analysis, one can use the original tweets (from their respective languages) without any preprocessing, and it is then possible to classify sentences that contain words from multiple languages at once. For example, a Spanish tweet that contains English cursing can be easily classified, which is not true when training independent per-language classifiers.

A recent but widely-used approach for modeling sentiment classifiers is based on Recurrent Neural Networks (RNNs), which process text encoded as a sequence of word embeddings (Γ). RNNs are networks that learn recurrent weight matrices for understanding temporal relationships among text of variable size. This flexibility is quite attractive for text-based learning such as sentiment analysis. LSTMs (Long Short-term Memory) [33] are more complex RNN-based architectures that are capable of learning long-term dependencies and forgetting useless information within a given sentence. The baseline LSTM approach that we employ in this paper for multilingual sentiment analysis is hereby called as **LSTM-Emb**.

A more recent approach [34] employs a simple convolutional layer together with a max-pooling-over-time operation instead of LSTMs for learning over text. Such architecture has the advantage of being faster than RNNs while providing similar predictive performance. Formally, the convolutional layer convolves Γ with f filters $3 \times d$ resulting in feature maps $\mathcal{M} \in \mathbb{R}^{f \times (n-2)}$. The max-pooling-over-time layer is responsible for selecting the most relevant features within the temporal dimension by using filters of size $1 \times (n-2)$. The resulting representation is $\mathcal{R} \in \mathbb{R}^{f \times 1}$, which is linearly mapped to the C classes. In the context of Twitter sentiment analysis, classes can be the polarities *positive* and *negative*, configuring a softmax output with two neurons. The baseline models that are based in this architecture of convolutional network are hereby referred as **Conv-Emb**.

B. Character-level Embedding

An alternative way of embedding words into multidimensional spaces is through the use of its basic components, the characters [15]. In a character-based representation, all characters $c_i \in \{c_1, c_2, ..., c_m\}$ in $\mathcal{T}$ are mapped to a binary matrix of size $m \times \eta$, where $\mathcal{V}$ is the alphabet with η characters, as presented in Figure 2. This representation is based on a 2D fixed-size input tensor for encoding text. The original architecture proposed in [15], hereafter called **Conv-Char**, comprises several convolutional layers that act as an embedding learning step, with the advantage of requiring only a small alphabet in memory instead of a large vocabulary. Even though **Conv-Char** comprises more convolutional layers than **Conv-Emb**, it requires fewer parameters to be learned.


Alphabet
Text

	a	b	c	d	e	f	h	i	...	η
I	0	0	0	0	0	0	0	1	...	0
	0	0	0	0	0	0	0	0	...	0
c	0	0	1	0	0	0	0	0	...	0
a	1	0	0	0	0	0	0	0	...	0
n	0	0	0	0	0	0	0	0	...	0
	0	0	0	0	0	0	0	0	...	0
d	0	0	0	1	0	0	0	0	...	0
o	0	0	0	0	0	0	0	0	...	0
	0	0	0	0	0	0	0	0	...	0
i	0	0	0	0	0	0	0	1	...	0
t	0	0	0	0	0	0	0	0	...	0

Fig. 2. Character-level representation of sentence "I can do it".

In this paper, we propose a novel character-based language-agnostic architecture for sentiment analysis, namely **Conv-Char-S**, which borrows the advantages from both **Conv-Emb** and **Conv-Char**. The architecture contains an amount of parameters which can be from 6 up to 75 times smaller than **Conv-Char** (depending on the number of convolutional filters) without significant performance degradation.

Conv-Char-S is designed for solving the inherent problems of word-embedding and n-gram based approaches. The proposed architecture understands sentences by processing a character-level input. Figure 3 depicts the proposed approach. It comprises essentially three components: (i) a convolutional layer with f receptive fields $7 \times \eta$; (ii) one max-pooling-over-time layer for selecting the most important feature throughout the sentence; and (iii) a final fully-connected layer that performs the linear mapping of pooled values into class scores.

The convolutional layer is responsible for understanding the input at character level. It can be seen as a feature creation procedure with a learned preprocessing step that generates a proper sentence-level embedding. In this step, the network is capable of correcting typos, understanding relationships between words, characters, and signals present in the known vocabulary. The max-pooling-over-time is inspired by the work

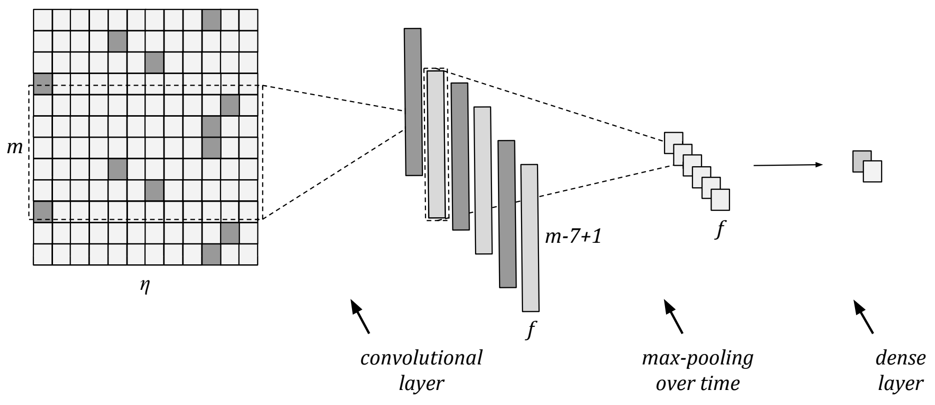


Fig. 3. Architectures based on character-level embedding: **Conv-Char** [15] and **Conv-Char-R** (ours).

TABLE I. AMOUNT OF PARAMETERS (CONSIDERING WORD-EMBEDDING AS TRAINABLE PARAMETERS) AND MEMORY REQUIRED FOR STORING BOTH DATA AND MODEL FOR EACH METHOD.

Network	# Parameters	Memory
LSTM-Emb [33]	$\approx 49M$	$\approx 1.2GB$
Conv-Emb [34]	$\approx 47M$	$\approx 1GB$
Conv-Char [14]	$\approx 3M$	$\approx 0.30GB$
Conv-Char-S(64) [Ours]	$\approx 0.04M$	$\approx 0.23GB$
Conv-Char-S(128) [Ours]	$\approx 0.07M$	$\approx 0.23GB$
Conv-Char-S(256) [Ours]	$\approx 0.13M$	$\approx 0.23GB$
Conv-Char-S(512) [Ours]	$\approx 0.26M$	$\approx 0.30GB$
Conv-Char-S(1024) [Ours]	$\approx 0.53M$	$\approx 0.38GB$

of Kim [34], and helps in reducing the dimensionality of the tensors. It also generates a regularizing effect, considering that it drastically reduces the input of the dense layer.

Table I presents the amount of parameters and memory required for storing both model and data of each deep neural architecture that was discussed. Note that our proposed approach is substantially more efficient in terms of parameters and memory consumption than the other neural architectures. **Conv-Char-S** has up to $1,225$ fewer parameters than **LSTM-Emb**, and consumes up to 5 times less memory than the word-embedding models. We empirically show in the next sections that **Conv-Char-S** is not just much more efficient but also more effective than the baseline deep neural architectures, reaching the novel state-of-the-art in language-agnostic multilingual Twitter sentiment analysis.

V. EXPERIMENTAL SETUP

We make use of the Twitter corpora from [35] to evaluate the proposed architecture. It contains data from 13 European languages, around 1.6 million annotated tweets, which is by far the largest corpora made publicly available so far. All tweets have been manually labeled into three classes: *positive*, *neutral*, and *negative*. Due to the semantic and syntactic structural differences among the 13 languages, we only consider a subset of tweets from four specific languages: English, Spanish, Portuguese, and German. We also reduce the problem to binary classification, i.e., we discard all neutral tweets. Note that the dataset does not provide the tweet itself, but rather a URL that leads to the tweets, and thus some tweets may no longer be available. Statistics of this subset can be seen in Table II.

Regarding the baseline approaches, our method differs from them by the fact that we explore not two but four languages at the same time, and also that we do not rely on any machine translation strategy. In our experiments we explore balanced data across languages as presented in Table II, which goes against the typical constraint faced by previous CLSC systems in which there are scarce resources in the training language and a large amount of unlabeled data in the target language. Hence, we do not consider work on bilingual sentiment classification and similar approaches that rely on machine translation as a baseline method in the experimental analysis. We compare **Conv-Char-S** with **LSTM-Emb**, **Conv-Emb**, **Conv-Emb-Freeze** (a variant of **Conv-Emb** in which we do not update the word embedding), **Conv-Char**, and with a traditional SVM-based approach [36], [37].

For finding the proper number of convolutional filters f (or LSTM hidden layer neurons) and dropout γ, we performed a grid search by investigating the following values: $f \in \{64, 128, 256, 512, 640, 768, 1024\}$ and $\gamma \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$. Regarding **Conv-Emb**, we set the embedding size to $d = 300$ as in [34]. The word embedding are initialized with random values and then optimized through backpropagation during training. The vocabulary size is set to $157,000$ words, composed by words in the training set from the four languages. We use 128 convolutional filters and dropout of 0.8 in the final dense layer. The learning rate is set to 1×10^{-4}. The variant **Conv-Emb-Freeze** has the same parameters than **Conv-Emb** though the word embeddings are not updated throughout the training process. For **LSTM-Emb**, we use the default LSTM implementation with input and forget gates, though discarding peephole connections. The LSTM comprises a single hidden layer with 256 $tanh$ neurons. We set the embedding size to $d = 300$, and use dropout rate (γ) of 0.5 in the final layer. For **Conv-Char**, it takes as input $m = 140$ characters and accepts an alphabet size of $\eta = 73$. Given the large amount of parameters and its capability of quickly overfitting the training data, we apply dropout rates of 0.8 and 0.9 in the two final layers of the network, respectively. Learning rate is set to $\alpha = 10^{-3}$. **Conv-Char-S** has the same hyper-parameters, and dropout of 0.9 is applied over its single dense layer. For all neural networks, we employ the Adam update rule for optimization with mini-batches of 128 training instances. Since the convolutional architectures

TABLE II. TWEET CORPORA. EACH OF THE SETS CONTAINS POSITIVE AND NEGATIVE TWEETS FROM 4 LANGUAGES.

	Training		Validation		Test		
Language	Negative	Positive	Negative	Positive	Negative	Positive	Total
English	7,784	7,645	1,131	1,146	2,200	2,264	22,170
German	7,502	10,727	1,063	1,544	2,057	2,944	25,837
Portuguese	17,170	12,202	2,427	1,745	4,990	3,560	42,094
Spanish	8,211	18,491	1,189	2,574	2,372	5,251	38,088
Multilingual	40,667	49,065	5,810	7,009	11,619	14,019	128,189
Share of entire corpora	89,732 (70%)		12,819 (10%)		25,638 (20%)		

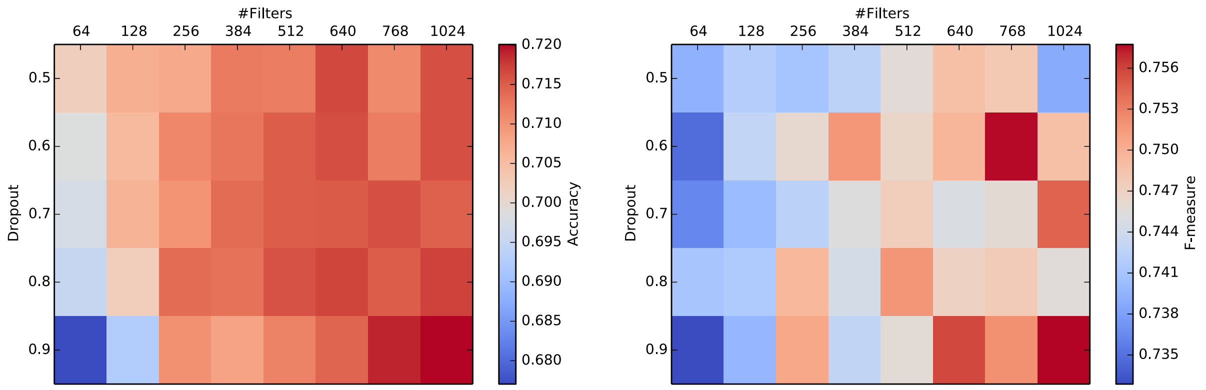


Fig. 4. Performance of Conv-Char-S by varying architectural choices. On the left the impact of dropout and number of filters on model accuracy. On the right, the impact of the same components on F-Measure.

comprise ReLU neurons, we initialize their weight matrices with the procedure described in [38]. We only use dropout to regularize dense layers. All models are trained with the negative log-likelihood loss function over a softmax layer that comprises two neurons (positive and negative). The SVM baseline employs a Gaussian kernel over the features, whose representation is based on unigrams, bigrams, and trigrams. We train an SVM model with a traditional pre-processing phase in NLP that consists of the following steps: normalization by setting all text to lowercase, stop-words and noise (URLs, @user and #) removal and stemming.

We evaluate the experiments with two traditional classification performance measures: *accuracy* and *F-Measure*. Accuracy is the rate of correctly classified instances whereas F-Measure is the harmonic mean of precision and recall. Both measures range within $[0, 1]$, with better scores indicated by higher values.

VI. RESULTS

In this section, we report several experiments for investigating: (a) the impact of architectural modifications in the predictive performance; (b) suitability of each method for learning language-agnostic features; (c) overfitting robustness; and (d) performance achieved by each methods.

We perform experiments for analyzing the relation between the number of convolutional filters, regularization, and predictive power of the proposed **Conv-Char-S** method. Such analysis allows to better understand the robustness of our architecture regarding the hyper-parameters' choices. Figure 4 depicts the relation between the number of convolutional filters *versus* the amount of required regularization. We can see that the number of filters is directly related to the accuracy of the model, achieving better results when using more filters. Values of F-Measure show the importance of dropout when training a large number of convolutional filters. Overall, models with ≥ 512 convolutional filters provide better predictive performance, although the differences are not significant, not surpassing 3% (varying between 0.73 and 0.76). Indeed, such architecture proved to be robust to severe hyper-parameter changes. Note that it achieves competitive results even with very reduced configurations, e.g., with 64 convolutional filters it requires $\approx 1225\times$ fewer parameters than the LSTM-based model and achieves similar results.

Figure 5 depicts both training and validation loss values throughout the optimization process. We show only the top three hyper-parameter configurations to better visualize the resulting behaviors. Note that the models trained with 90% of dropout need more epochs for convergence, despite being much more robust in terms of overfitting, whereas the model trained with 60% of dropout converges much faster, though quickly shows alarming signs of overfitting.

A. Conv-Char-S vs. Baselines

We compare the four different deep neural network architectures, as well as SVM-based approaches regarding their predictive performance in the multilingual Twitter dataset: **LSTM-Emb**, **Conv-Emb[-Freeze]**, **Conv-Char**, **Conv-Char-S**, and **SVM-[U,B,T]**. For the SVM-based models, we evaluate the performance of using only unigrams (SVM-U), only bigrams (SVM-B), only trigrams (SVM-T), and combinations among them (SVM-UB, SVM-UBT). Results of accuracy and

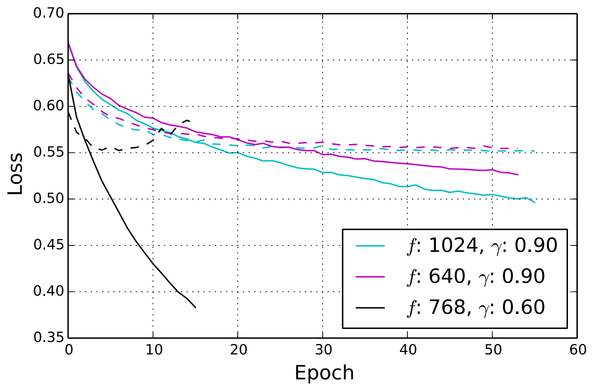


Fig. 5. Training and validation loss per epoch. Continuous lines depict values of training loss. Dotted lines indicate validation loss.

F-Measure are presented in Table III. We also show the per-language results for the neural architectures, so we can evaluate if there is significant variation of results in a language basis (see Tables IV and V).

TABLE III. ACCURACY AND F-MEASURE VALUES.

Method	Accuracy	F-Measure
SVM-U	0.694	0.751
SVM-UB	0.672	0.746
SVM-UBT	0.660	0.743
SVM-B	0.562	0.717
SVM-T	0.556	0.712
Conv-Emb-Freeze	0.688	0.728
Conv-Emb	0.715	0.747
LSTM-Emb	0.711	0.752
Conv-Char	0.696	0.721
Conv-Char-S [ours]	**0.720**	**0.756**

Note that our method (Conv-Char-S) achieves both the best F-Measure (0.756) and accuracy (0.720) values, followed by **LSTM-Emb** and **Conv-Emb**. Also, observe that the SVM-based models with n-grams are quite competitive, although requiring an extensive preprocessing phase. The SVM-based models are a strong baseline much due to the cleaning preprocessing techniques (e.g., stopwords removal and stemming) applied over the input data, together with their ability in handling high-dimensional feature spaces even for medium-sized datasets. Note that for deploying an SVM for sentiment analysis, one needs access to the stopwords and stemmers for each individual language, which can be time-demanding and costly when handling several languages in the dataset. Also, recall that the unigram feature representation tends to be very sparse when adding novel languages, which can be particularly problematic having in mind the *curse of dimensionality*.

Regarding the convolutional models that employ different text representations, one can observe that word-embedding models seem to perform consistently better when updating the embeddings. Note that learning the embedding through backpropagation is always better than freezing the embedding and then only learning the remaining parameters. As presented in Table I, the excessively large amount of parameters to be learned in word-embedding-based models is a prominent disadvantage when adopting this representation. The character-based models are much cheaper, and even though **Conv-Char** is not capable of outperforming the word-embedding architectures, one can argue it still presents a good trade-off between predictive power and model complexity. In addition, typos and words slightly different require new vectors for the embedding of word-based models, demanding lots of memory for storing such large vocabulary. The same is not true for character-based models, which do not rely on any fixed-size word list.

TABLE IV. PER-LANGUAGE F-MEASURE.

Method	English	German	Portuguese	Spanish
Conv-Emb-Freeze	0.736	0.752	0.583	0.795
Conv-Emb	0.763	0.782	0.608	0.804
LSTM-Emb	**0.779**	0.783	0.619	**0.806**
Conv-Char	0.729	0.758	0.589	0.776
Conv-Char-S [ours]	0.771	**0.797**	**0.647**	0.799

TABLE V. PER-LANGUAGE ACCURACY VALUES.

Method	English	German	Portuguese	Spanish
Conv-Emb-Freeze	0.734	0.694	0.671	0.678
Conv-Emb	0.766	0.733	0.695	0.693
LSTM-Emb	**0.770**	0.734	0.686	**0.697**
Conv-Char	0.736	0.718	0.688	0.668
Conv-Char-S [ours]	0.762	**0.755**	**0.706**	0.694

Regarding our proposed approach, **Conv-Char-S**, one can see that it presents much better results than its counterpart character-based network, **Conv-Char** [14]. **Conv-Char-S** arguably presents the best results in terms of predictive performance and amount of parameters among all neural architectures. Indeed, **Conv-Char-S** in its larger version presents results roughly 5% better than **Conv-Char** in terms of accuracy and F-Measure, despite having fewer parameters. By analyzing Tables V and IV, we can observe that our approach is consistently better for German and Portuguese, which seem to be the more complicated languages to learn, and present virtually the same results for English and Spanish.

Finally, we argue on the advantages of the multilingual strategy rather than using one classifier per language. For the latter, one would need to train a prior classifier that first detects the language of the tweet and then redirects it to the proper per-language classifier. This cascade of classifiers enhances the cost of the classification process and may eventually deteriorate the final results, since the incorrect identification of the language will harm the chances of correctly classifying the sentiment of the tweet. Hence, our multilingual approach seems to be much more reasonable, since it uses a single classifier that can understand any of the languages that were used during training. Another advantage of the multilingual setup is that the tweet to be analyzed can contain terms from distinct languages without affecting the classification process, which is not true for the per-language classification approach.

VII. NETWORK VISUALIZATION

For visualizing what was really learned by our approach, we propose a visualization scheme based on [39], which presents a visualization strategy for image-based Convolutional Networks. By employing such a strategy, we aim at discovering

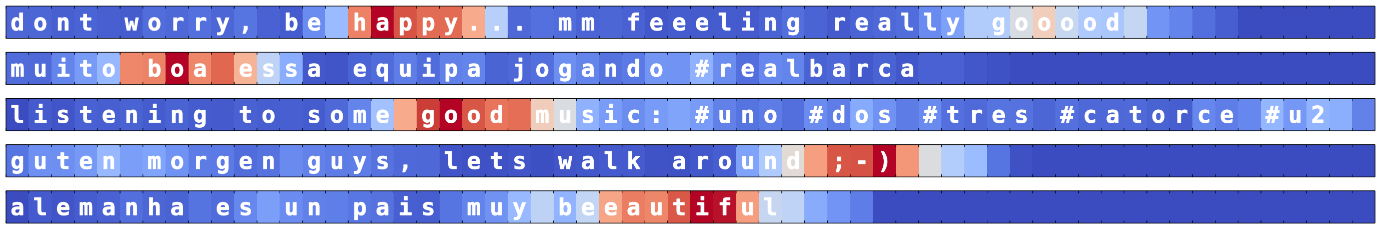


Fig. 6. Character-level visualization of the most relevant input generated via backpropagated gradients. All sentences above are of positive polarity. Best visualized in colors. Red means greater influence in the sentiment classification, whereas blue tones are less significant.

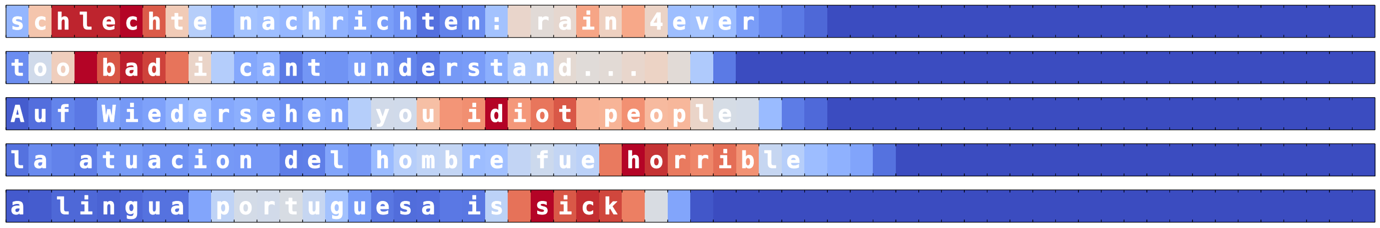


Fig. 7. Character-level visualization of sentences of negative polarity. Best visualized in colors. Red means greater influence in the sentiment classification, whereas blue tones are less significant.

whether the network learns meaningful interpretable information. Note that our network performs text understanding from scratch by analyzing all characters without any preprocessing. We propose the visualization as follows. A single weight update of a weight matrix Θ through vanilla stochastic gradient descent is given by

$$\Theta \leftarrow \Theta - \alpha \frac{\partial \mathcal{E}(\hat{y}, \mathcal{P})}{\partial \Theta}. \tag{1}$$

Intuitively, gradients with respect to a given class carry information regarding the updates needed for each weight in the model. In a similar fashion, by backpropagating gradients back to the input, we learn the importance of each input feature in performing the classification. In this step, let $\mathcal{D}$ be the final dense layer, $\mathcal{S}$ the max-over-time-pooling, $\mathcal{G}$ the convolutional layer, and finally Γ_i is an input instance encoded at character level. Thus, we want to evaluate the gradients $\hat{\Gamma}$ of the input which is given by:

$$\hat{\Gamma} = \frac{\partial \mathcal{E}(\hat{y}, \mathcal{P})}{\partial \Gamma_i} = \frac{\partial \mathcal{G}}{\partial \Gamma_i} \frac{\partial \mathcal{S}}{\partial \mathcal{G}} \frac{\partial \mathcal{D}}{\partial \mathcal{S}} \frac{\partial \mathcal{E}(\hat{y}, \mathcal{P})}{\partial \mathcal{D}} \tag{2}$$

For generating the visualization, we discard the negative gradients, keeping only information that positively impacts for minimizing the loss function. We also normalize gradient values by setting the maximum value at 1. Note, however, that $\hat{\Gamma}$ is somehow the character-level input reconstruction, which is a matrix $m \times \eta$, and we want to visualize the impact of each character in the provided classification, which requires a vector. Now, let $\mathcal{I} \in \mathbb{R}^{1 \times m}$ be the character-level text reconstruction generated by the gradients magnitude, which is finally generated by:

$$\mathcal{I}_j = \sum_{i}^{\eta} max(\hat{\Gamma}_{ji}, 0) \tag{3}$$

Figures 6 and 7 depicts several scenarios generated by using this strategy. Our findings based on this qualitative analysis are as follows: (i)our character-level based network is capable of identifying the most important words within sentences. This is true even for typos, which are particularly problematic in word-embedding and n-grams approaches; (ii) in particular cases, the network itself learns the most important parts of the words (and sentences). Some cases seem to mimic the stemming preprocessing step; (iii) our approach can leverage *emoticons* information such as :) and $: D$, as well of prolonged words such as *gooood*; and (iv) the proposed network does properly identify polarity within sentences even when facing words from multiple languages together.

VIII. Conclusions

In this paper, we proposed a novel and efficient neural architecture for performing language-agnostic sentiment analysis over tweets in four languages. To the best of our knowledge, this is the first paper to investigate character-based neural networks for language-agnostic translation-free sentiment classification in multilingual scenarios. Our approach does not rely on machine translation, paired datasets, nor word-embeddings to perform the proposed task. We have demonstrated that our method outperforms all other proposed deep architectures and also traditional SVM-based approaches, achieving state-of-the-art results in the proposed task. Moreover, our method is far more efficient than the second-best baseline, requiring up to 1225$\times$ fewer parameters in its smaller version while not requiring any kind of preprocessing whatsoever. Our approach generates models that are easily implemented and require half a megabyte for storage. As a final contribution, we proposed a novel method for visualizing and understanding character-level networks. Our models proved to be reasonable and interpretable even when dealing with sentences whose words come from multiple distinct languages. As future work, we intend to build a larger corpora of tweets and reviews that spans more languages, and also to explore tasks other than sentiment analysis.

IX. Acknowledgments

The authors would like to thank FAPERGS, CAPES, CNPq, Motorola Mobility, and Google for funding this research.

References

[1] E. Martínez-Cámara, M. T. Martín-Valdivia, L. A. Urena-López, and A. R. Montejo-Ráez, "Sentiment analysis in twitter," *Natural Language Engineering*, vol. 20, no. 01, pp. 1–28, 2014.

[2] J. S. Olsson, D. W. Oard, and J. Hajič, "Cross-language text classification," in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 2005, pp. 645–646.

[3] B. Fortuna and J. Shawe-Taylor, "The use of machine translation tools for cross-lingual text mining," in *Proceedings of the ICML Workshop on Learning with Multiple Views*, 2005.

[4] C. Banea, R. Mihalcea, J. Wiebe, and S. Hassan, "Multilingual subjectivity analysis using machine translation," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2008, pp. 127–135.

[5] J. G. Shanahan, G. Grefenstette, Y. Qu, and D. A. Evans, "Mining multilingual opinions through classification and translation," in *Proceedings of AAAI Spring Symposium on Exploring Attitude and Affect in Text*, 2004.

[6] G. Zhou, Z. Zeng, J. X. Huang, and T. He, "Transfer learning for cross-lingual sentiment classification with weakly shared deep neural networks," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2016, pp. 245–254.

[7] X. Tang and X. Wan, "Learning bilingual embedding model for cross-language sentiment classification," in *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2014 IEEE/WIC/ACM International Joint Conferences on*, vol. 2. IEEE, 2014, pp. 134–141.

[8] Y. Guo and M. Xiao, "Cross language text classification via subspace co-regularized multi-view learning," *arXiv preprint arXiv:1206.6481*, 2012.

[9] K. Duh, A. Fujino, and M. Nagata, "Is machine translation ripe for cross-lingual sentiment classification?" in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, 2011, pp. 429–433.

[10] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *JMLR*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[11] 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)*, vol. 1631, 2013, p. 1642.

[12] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 160–167.

[13] J. Lee, K. Cho, and T. Hofmann, "Fully character-level neural machine translation without explicit segmentation," *arXiv preprint arXiv:1610.03017*, 2016.

[14] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in Neural Information Processing Systems*, 2015, pp. 649–657.

[15] X. Zhang and Y. LeCun, "Text understanding from scratch," *arXiv preprint arXiv:1502.01710*, 2015.

[16] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1422–1432.

[17] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, no. Aug, pp. 2493–2537, 2011.

[18] H. Zhou, L. Chen, F. Shi, and D. Huang, "Learning bilingual sentiment word embeddings for cross-language sentiment classification." ACL, 2015.

[19] D. Maynard and A. Funk, "Automatic detection of political opinions in tweets," in *Extended Semantic Web Conference*, 2011, pp. 88–99.

[20] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014.

[21] M. Gamon, A. Aue, S. Corston-Oliver, and E. Ringger, "Pulse: Mining customer opinions from free text," in *International Symposium on Intelligent Data Analysis*, 2005, pp. 121–132.

[22] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proceedings of the 42nd annual meeting on Association for Computational Linguistics*, 2004, p. 271.

[23] L. Shi, R. Mihalcea, and M. Tian, "Cross language text classification by model translation and semi-supervised learning," in *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, 2010, pp. 1057–1067.

[24] R. Mihalcea, C. Banea, and J. M. Wiebe, "Learning multilingual subjective language via cross-lingual projections," 2007.

[25] X. Zhou, X. Wan, and J. Xiao, "Attention-based lstm network for cross-lingual sentiment classification."

[26] ——, "Cross-lingual sentiment classification with bilingual document representation learning."

[27] H. Pham, M.-T. Luong, and C. D. Manning, "Learning distributed representations for multilingual text sequences," in *Proceedings of NAACL-HLT*, 2015, pp. 88–94.

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

[29] K. Dave, S. Lawrence, and D. M. Pennock, "Mining the peanut gallery: Opinion extraction and semantic classification of product reviews," in *Proceedings of the 12th international conference on World Wide Web*, 2003, pp. 519–528.

[30] J. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin, "Learning subjective language," *Computational linguistics*, vol. 30, no. 3, pp. 277–308, 2004.

[31] A. Abbasi, H. Chen, and A. Salem, "Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums," *ACM Transactions on Information Systems (TOIS)*, vol. 26, no. 3, p. 12, 2008.

[32] C. N. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts." in *COLING*, 2014, pp. 69–78.

[33] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[34] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[35] I. Mozetič, M. Grčar, and J. Smailović, "Multilingual twitter sentiment classification: The role of human annotators," *PloS one*, vol. 11, no. 5, p. e0155036, 2016.

[36] V. Vapnik and C. Cortes, "Support vector networks," *Machine Learning*, vol. 20, p. 273, 1995.

[37] V. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, pp. 988–999, 1999.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.

[39] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.