Towards better Sentence Classification for Morphologically Rich Languages

by

Madhuri Tummalapalli, Manoj Chinnakotla, Radhika Mamidi

in

19th International Conference on Computational Linguistics and Intelligent Text Processing (*CICLing-2018*)

Hanoi, Vietnam

Report No: IIIT/TR/2018/-1

Centre for Language Technologies Research Centre
International Institute of Information Technology
Hyderabad - 500 032, INDIA
March 2018

Towards better Sentence Classification for Morphologically Rich Languages

Madhuri Tummalapalli[1], Manoj Chinnakotla[2], and Radhika Mamidi[1]

[1] Language Technologies Research Center,
KCIS, IIIT Hyderabad
madhuri.tummalapalli@research.iiit.ac.in,
radhika.mamidi@iiit.ac.in
[2] Microsoft, India
manojc@microsoft.com

Abstract. Many methods have been developed for various sentence classification tasks for English, which usually exploit linguistic resources like parsers or rely on the large amount of annotated or unannotated data, making it difficult to adapt them to other languages. In this paper, we present an evaluation of popular deep learning methods for sentence classification on the morphologically rich Indian languages, specifically, Hindi and Telugu. For this purpose, we also created a question classification dataset for Hindi, by translating the TREC-UIUC dataset. We show that character based input can enhance the performance of current classification systems for morphologically rich languages. Finally, we show that our *multiInput-CNN* variant is able to perform better than our baselines in two out of three tasks in Hindi and Telugu, while giving comparable results for others.

Key words: Text Classification, Language Independence, Resource Scarce Languages, Morphologically Rich Languages, Indian Languages

1 Introduction

The Indian subcontinent has more than 120 languages out of which 29 languages are spoken by more than a million people each[3]. Most of these languages, are however, ill represented as there are a very few resources, systems and techniques that have been developed for them. There is therefore a need for the development of NLP techniques in these languages.

In order to cut down the cost of building end to end systems for individual languages, we need to develop techniques that do not rely on any expensive language resources like parsers or be specific to the structure of any single language.

Traditionally, systems used to be developed for specific tasks and languages, wherein carefully designed feature sets were used for various classification tasks like Sentiment Analysis, Question Classification etc. These features exploited the language structures, often captured through parse trees [1, 2]. In some other

[3] https://www.mapsofindia.com/culture/indian-languages.html

cases, like Sentiment Analysis, resources specific to the task, such as sentiment lexicons were developed [3, 4]. Although, these systems work very well for their specific tasks and languages, such hand crafted features and resources make it difficult to adapt them to other tasks or languages.

Recently, deep learning techniques have made it possible to develop systems that are more generic in the set of tasks they can handle as they do not require hand-crafted features. These deep learning systems have been developed for various sentence classification tasks and rely majorly on the availability of large amount of labeled or unlabeled data. Such systems are therefore attractive when developing language independent systems.

In several instances, classification in resource scarce languages is dealt with by translating the data to English or other resource full languages, and then applying existing models for classification. But, the noise induced during translation hinders the efficiency of these systems as we show later.

In this paper, we present an evaluation of the existing deep learning models for two tasks - Sentiment Analysis and Question Classification in three languages: English, Hindi and Telugu. These three languages differ in their linguistic structure and morphology. While, English is a fixed word order language, Hindi and Telugu are free word order languages. The morphological richness of the languages also increases from English to Hindi (an Indo-Aryan language) and to Telugu (a Dravidian language).

We also compare the performance of word and character based ngrams as input, and find that as a language becomes morphologically richer, character based input performs better. We show in our final model, *multiInput-CNN*, that a combination of word and character based input is able to perform better than our baselines in two out of three tasks in Hindi and Telugu, while giving comparable results for others.

Our main contributions through this paper are as follows:

- An evaluation of popular deep learning methods for Sentence Classification, while paying special attention to their performance on morphologically rich languages. Evaluations are done on English, Hindi and Telugu.
- Showing the importance of character based ngrams for sentence classification in morphologically rich languages.
- *multiInput-CNN*, a CNN variant, that takes both word and character based embeddings as input.
- A Hindi Question Classification dataset created by translating TREC-UIUC English dataset into Hindi.

2 Related works

In recent years, Neural Networks have been used in a variety of tasks in Natural Language Processing (NLP). A particularly influential work has been [5], where a neural network with just one layer of convolution, with word vectors (random or word2vec [6] vectors) as input was shown to be effective on seven sentence classification tasks, out of which in 4 tasks, state-of the art was achieved. A variant of CNN called Dynamic CNN was proposed in another work, where the idea of dynamic k-max pooling was introduced [7].

Other works using recursive or recurrent neural models include [8–12]. While [12] apply an LSTM on the sequence of words in a sentence, [8, 10, 11] exploit the phrase level information in their respective datasets, to recursively find the sentiment of the entire sentence. [9, 13] depends on the usage of a dependency parser for sentence classification. Since our aim is to build a system that uses minimal language-specific resource, we do not include these methods in our study.

[14] use CNN and utilize word order information for classification. [15] uses an LSTM-CNN network for a dependency sensitive model. Use of LSTMs and RNNs can have the disadvantage of not being flexible towards free word order languages, as we show later in our experiments with LSTMs and Bi-LSTMs.

Attention based models have also been used for sentence classification. [16] uses a CNN-RNN hybrid attention based model for phrase aware classification. Similar is the case with attention networks in [17–19]. Whereas, [20] propose the use of very deep CNN for classification. As we move towards such deeper models, the number of parameters to be learned by the network increases exponentially, which in turn requires large datasets. Thus, making it inapplicable to low resource languages where large datasets are difficult to find.

Some variations of CNN have also been tried. Many of these try to modify or improve the input (usually word vectors) to enable the network to learn better [21–24]. [25], bring in topic models for disambiguation of polysemic words when using word embeddings. Whereas, [26] try to find a better initialization for the convolution filters.

[27–30] use character based inputs for CNN. [27, 28, 30] use very deep CNNs to extract information from character level input. Though, [29] on the other hand show that a single convolution with character embedding is good enough for classification purposes. These works only use character 1-grams as input, we demonstrate later that character n-gram input can lead to much better performance in morphologically rich languages.

Proposing a yet another type of variance, [31, 32] modify the CNN to account for multiple sets of word embeddings as input. While, [31] combine the embeddings at convolution stage, [32] perform independent convolution operations for all sets, and combine them at the penultimate layer.

Some cross-lingual techniques have been explored which use the resources in resource-rich languages for classification in resource-scarce languages. [33] is able to use weakly supervised data for training their network, which is based on the usage of emoticons in Twitter, while [34] translate their data word-to-word and then augment the training with polarity words to achieve better results. [35, 36] use non-deep learning approaches to solve the problem. [37] requires a parallel corpus for its method, while [38] uses Wikipedia titles to construct a semantic space for multiple languages. Some other works like [39–41] rely on machine translation in their techniques, which, as we show later, induces a lot of noise into the data.

Few works have been done in Hindi and Telugu Sentiment analysis [34, 42]. While, [34] use an approach that is specific to sentiment analysis, [42] evaluate various non-deep learning approaches for classification.

In this paper, we present our evaluations on deep learning methods for language independent Sentence Classification, paying special attention to the case of morphologically rich languages. Thus, we present our evaluations in Hindi and Telugu along with English, using CNNs, LSTMs and Bi-LSTMs, restricting ourselves to the usage of minimum language resources.

3 Datasets

For the purpose of our evaluations, we use five datasets. Three of which are for Sentiment Analysis, one each for English, Hindi and Telugu. The English dataset consists of only two classes (positive and negative), the Hindi and Telugu datasets consist of three classes (positive, negative and neutral) each. The other two datasets are for Question Classification in English and Hindi. The English dataset is the TREC-UIUC[4] dataset. The Hindi dataset has been created by the authors by translating the English TREC-UIUC dataset. Specific details about each dataset have been mentioned in the following sub-sections.

3.1 Question Classification Datasets

TREC-UIUC dataset (TREC-En) This dataset was released by [43] along with an answer type taxonomy containing 6 core classes and 50 fine classes. The dataset is divided into two sets of annotated questions, a training set with 5452 questions and a test set with 500 questions. The 6 core classes along with the number of instances in each have been listed in Table 1.

TREC Hindi dataset (TREC-Hi) We created a new dataset for Question Classification in Hindi by translating the TREC-UIUC dataset described above. The TREC-UIUC dataset was first translated to Hindi with the help of Google Translate[5], and then was manually validated and corrected by the authors. A few sentences where the translations were not meaningful due to cultural differences, were omitted from the train and test sets. The final train and test sets have 5444 and 499 questions respectively.

To validate the dataset, we gave 25 randomly selected sentences each from the translated set to 6 native speakers of Hindi, who rated them according to whether the sentences were completely incorrect (score 1) or completely correct and natural (score 5). We got an average score of 4.16 from our evaluations. The dataset statistics have been mentioned in Table 1.

[4] `http://cogcomp.cs.illinois.edu/Data/QA/QC/`
[5] `https://translate.google.com`

Table 1. Statistics for Question Classification datasets

Abbreviation	Class	TREC-En		TREC-Hi	
		Train	Test	Train	Test
DESC	Description	1162	138	1162	138
ENTY	Entity	1250	94	1248	93
ABBR	Abbreviation	86	9	86	9
NUM	Number	1223	65	1219	65
HUM	Human	896	113	895	113
LOC	Location	835	81	834	81

Table 2. Statistics for Sentiment Analysis datasets

Dataset	Positive	Neutral	Negative
MR	5331	-	5331
Senti-Hindi	2240	3408	932
Senti-Telugu	1491	2477	1441

3.2 Sentiment Analysis Datasets

Table 2 contains statistics for all the sentiment analysis datasets, while their details have been listed in the following sub-sections. As none of the datasets contains a train-test split, we perform 10-fold cross validation to get results for these datasets.

MR dataset The dataset contains 10662 movie reviews, one sentence each, in English [44]. The sentences are labeled for positive or negative sentiment.

Sentiment Analysis-Hindi (Senti-Hi) This dataset also contains movie reviews and has three classes: positive, negative and neutral. There are a total of 6580 sentences.

Sentiment Analysis-Telugu (Senti-Te) The Telugu sentiment analysis dataset contains 5409 sentences annotated for positive/negative/neutral sentiment [45].

4 Models Used

In this section, we describe the different types of inputs and classification techniques used in our experiments.

4.1 Input

Sequence of Words Here, a sentence is represented as a sequence of words. Each word in turn, is represented by a vector of fixed dimension d. The vector might be:

- A random vector (**word-rand**)
- Pre-trained word2vec vector [6]. The word vectors were obtained by training on Wikipedia dumps[6] for all languages. (**word-word2vec**)

Sequence of characters (char-1-gram) Here, a sentence is represented as a sequence of characters. Each character in turn, is represented by a random vector of fixed dimension d.

Sequence of character n-grams (char-ngram) Here, a sentence is represented as a sequence of character n-grams (which do not span across words), where each character-ngram is represented by a random vector of fixed dimension d. The idea behind using character n-grams is that they can help us represent the sentence as a sequence of morphemes, which are the smallest meaningful unit in a language. This, in turn can be beneficial for morphologically rich languages, where each word contains substantial information often due to the number of morphemes present in it. In the set of languages chosen by us, the morphological richness increases from English to Hindi and to Telugu. The sequence of characters input described previously, is a special case when $n = 1$. To the best of our knowledge, character n-grams, when $n > 1$, have not been used for the task of sentence classification before.

4.2 CNN

We borrow our CNN architecture from [5]. [5] defines a number of different CNN architectures. Of these, we show our results on CNN-rand and CNN-non-static. We also did our experiments with CNN-static, but as CNN-non-static always performed better, we do not mention those results here.

4.3 LSTM and Bi-LSTM

The LSTM architecture was introduced by [46]. For our experiments, we apply a vanilla LSTM (implemented in Keras [47]) over different representation of sentences as described in 4.1.

The Bi-LSTM architecture is a variant of LSTM and was introduced by [48]. A Bi-direction LSTM consists of two LSTMs, one in which the input is processed from beginning of the sequence to the end, whereas in the other it is processed backward - from the end to the beginning of the sequence. The outputs from both the networks is concatenated to give the final output.

[6] `https://dumps.wikimedia.org/`

In our experiments, the outputs from the LSTM or Bi-LSTM are fed to a fully connected layer, with *softmax* activation and *categorical-crossentropy* loss for classification.

4.4 Our MultiInput-CNN variant

[5] had also proposed a multichannel variant of their basic CNN architecture called CNN-multichannel, where there were two input channels, both initialized with pre-trained word vectors. The same convolution filters were applied on both the channels. While the weights in one of the channels was updated, the other was kept static. [31] use a similar idea of multichannel input, but initialize each channel with a different set of pre-trained word embeddings. While [32] also uses multiple pre-trained word embeddings as inputs, but the convolution filters are applied independently to all the inputs, and the resultant output is joined at the penultimate layer. Applying an independent convolution on each input gives the freedom of finding and using the best filter dimensions and other parameters for each input.

We propose the use of both word and character based inputs in a CNN based model. The two inputs specifically are character n-gram embeddings and pre-trained word2vec word embeddings. We apply the complete CNN architecture as described as CNN-rand and CNN-non-static in [5] on both the inputs respectively. We then take an average of the outputs of the final (*softmax*) layers of the individual networks to get our final output. This network is also trained with a *categorical-crossentropy* loss.

This network, in our opinion is language independent, as it is able to leverage upon the morphological information with the help of character n-grams that is beneficial for morphologically rich languages, while the information captured in the rich word embeddings is retained.

4.5 SVM

We also experiment with SVM, as it enables us to get an insight as to when and how these neural architectures might be useful. The SVM is trained with a linear kernel on bag-of-words, bag-of-word ngrams and bag-of-character ngram input. We also feed a combination of word and character ngrams to the SVM for our experiments.

5 Experiments and Results

The experiments have been divided into three parts. Section 5.1 presents detailed evaluation and discussion of different baseline models and inputs on all datasets. Section 5.2 compares our *multiInput-CNN* model's performance with the baselines and other state-of-the arts. In Section 5.3, we try to understand the inherent dataset biases across languages.

Table 3. Evaluation of baseline models. All the results mentioned are percentage accuracies and the highest accuracy for each dataset in each model is written in **bold**.

Model	Input	MR	Senti-Hi	Senti-Te	TREC-En	TREC-Hi
CNN	word-rand	76.35	70.44	52.18	92.20	86.97
	word-word2vec	**81.04**	**73.67**	55.79	**94.00**	**93.19**
	char-1-gram	66.38	67.51	51.70	76.2	75.95
	char-ngram	77.40	72.06	**58.05**	89.20	88.18
LSTM	word-rand	**75.41**	**56.83**	45.82	**84.00**	**81.76**
	word-word2vec	71.68	51.81	45.79	82.60	81.56
	char-ngram	74.63	51.85	**45.83**	83.40	78.76
Bi-LSTM	word-rand	**75.19**	**68.65**	**52.30**	**88.80**	**84.17**
	word-word2vec	75.14	68.14	50.78	86.80	**84.17**
	char-ngram	74.91	68.18	50.99	82.20	82.57
SVM	word ngrams	75.2	71.00	53.95	90.00	87.17
	char ngrams	77.07	70.91	57.00	**91.20**	**87.58**
	(word+char) ngrams	**77.45**	**71.28**	**57.18**	**91.20**	87.17

5.1 Evaluation of Baseline Models

For the purpose of evaluating our baseline models (CNN, LSTM, Bi-LSTM, SVM) described in Section 4, we conducted a number of experiments. Parameter tuning was done for each model and each dataset, and the results have been presented in Table 3.

As we can see from the Table 3, character 1-grams input in CNN perform consistently bad for all the datasets. Perhaps, the reason being that our network is not deep enough to be able to process the character 1-grams and draw useful information from them.

Hindi and Telugu are free word order languages, and hence their performance suffers in sequential models like LSTM and Bi-LSTM, as can be seen from the results of Senti-Hi, Senti-Te and TREC-Hi. Although, there is a drop in the performance of LSTM and Bi-LSTM for English datasets as well, this drop is comparatively lesser than the other two languages. This drop also signifies that LSTM and Bi-LSTM models are not as efficient as CNNs for our task.

It is also essential to note that as we move towards morphologically richer languages, it is the character ngram inputs that lead to better results. This can be seen by observing the inputs at which Telugu dataset Senti-Te obtains highest accuracy in all the models. The only model where this trend fails is the Bi-LSTM model, where the difference in the highest accuracy and the one obtained when using character ngrams is very small.

Since our CNN model has just one layer of convolution, and used *global max pooling* instead of *local max pooling*, the model essentially picks the most salient n-grams from the input. This idea is confirmed from the results where, the SVM results for all languages can be seen to be roughly equivalent to the ones obtained with CNN model with word-rand input. Which also means, that the benefit of

Table 4. Comparison of *multiInput-CNN* with the state-of-the arts

Model	MR	Sent-Hi	Sent-Te	TREC-En	TREC-Hi
multiInput-CNN	81.28	**74.16**	**59.32**	94.20	91.58
CNN word-word2vec (CNN-non-static in [5])	81.29	73.67	55.79	94.00	**93.19**
CNN char-1-gram (similar to [29])	66.38	67.51	51.70	76.2	75.95
CNN char-ngram	77.40	72.06	58.05	89.20	88.18
[5]	81.5	-	-	92.8	-
[21]	77.77	-	-	92.6	-
[26]	82.1	-	-	94.4	-
[25]	**83**	-	-	84.1	-
[15]	82.2	-	-	85.6	-
[32]	-	-	-	95.52	-
[22]	-	-	-	**95.8**	-

Table 5. CNN Results on Translated datasets

Data	Original	English	Hindi	Telugu
MR	81.04	-	73.20	71.21
Senti-Hi	73.23	71.65	-	69.52
Senti-Te	58.05	51.57	51.54	-

using a CNN lies in the ability to leverage the information in pre-trained word embeddings like word2vec.

The results obtained from SVM when using both word and character level ngrams as input, and the results obtained with character ngrams as input in CNNs lead us to our final model, *multiInput-CNN*, where we combine character ngrams and pre-trained word embedding inputs.

5.2 multiInput-CNN

This model combines character ngrams and pre-trained word embedding inputs. While the character level inputs capture important information for morphologically rich languages, pre-trained word embedding inputs enable the model to capture important contextual information. The results of this model in comparison with the state-of-the arts have been shown in Table 4.

As we can see, the *multiInput-CNN* model is able to achieve better results than all our baselines for 2 out of 3 datasets in Hindi and Telugu, while obtaining results comparable to the state-of-the arts for English. For all the datasets, the word embeddings used were trained using word2vec on Wikipedia dumps of the respective languages.

5.3 Experiments with Translated Datasets

To understand possible biases created due to different datasets, we repeated our baseline experiments with CNNs on datasets obtained by automatically translating each sentiment analysis dataset into the other two languages with the help of Google Translate. The results showed that in general, the translated data always obtained a lower accuracy than the original data, signifying that translating data into a language like English and then classifying in that language can induce more errors. These results have been listed in Table 5

These results also help us establish that the Hindi and Telugu sentiment analysis datasets, are inherently much difficult as compared to the English dataset, and the difference in accuracies between the datasets may not be a reflection of difficulty of classification in that particular language only. Thus, throughout our paper, we have only focused on the trends in the performance of different models within a single dataset.

6 Conclusion and Future Work

Through our experiments with different deep learning models with different types of inputs, we showed that as we move to morphologically rich languages, character ngram based inputs lead to better performance than other input methods for most of the models. This conclusion also enabled us to present a model, *multiInput-CNN* that capitalizes on both word and character based inputs to give us a language independent model. While the character based embeddings provide morphological information to the model, pre-trained word embeddings carry contextual information with them. In the future, we can focus on better embeddings for character ngrams, that are able to capture morphological and contextual information.

References

1. Silva, J., Coheur, L., Mendes, A.C., Wichert, A.: From symbolic to sub-symbolic information in question classification. Artificial Intelligence Review **35** (2011) 137–154
2. Huang, Z., Thint, M., Qin, Z.: Question classification using head words and their hypernyms. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics (2008) 927–936
3. Kouloumpis, E., Wilson, T., Moore, J.D.: Twitter sentiment analysis: The good the bad and the omg! Icwsm **11** (2011) 164
4. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M.: Lexicon-based methods for sentiment analysis. Computational linguistics **37** (2011) 267–307
5. Kim, Y.: Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882 (2014)
6. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. (2013) 3111–3119

7. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188 (2014)
8. Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C.D., Ng, A., Potts, C.: Recursive deep models for semantic compositionality over a sentiment treebank. In: Proceedings of the 2013 conference on empirical methods in natural language processing. (2013) 1631–1642
9. Komninos, A., Manandhar, S.: Dependency based embeddings for sentence classification tasks. In: HLT-NAACL. (2016) 1490–1500
10. Dong, L., Wei, F., Zhou, M., Xu, K.: Adaptive multi-compositionality for recursive neural models with applications to sentiment analysis. In: AAAI. (2014) 1537–1543
11. Irsoy, O., Cardie, C.: Deep recursive neural networks for compositionality in language. In: Advances in neural information processing systems. (2014) 2096–2104
12. Hassan, A., Mahmood, A.: Deep learning for sentence classification. In: Systems, Applications and Technology Conference (LISAT), 2017 IEEE Long Island, IEEE (2017) 1–5
13. Xia, Y., Wei, Z., Liu, Y.: An efficient cross-lingual model for sentence classification using convolutional neural network. ACL 2016 (2016) 126
14. Johnson, R., Zhang, T.: Effective use of word order for text categorization with convolutional neural networks. arXiv preprint arXiv:1412.1058 (2014)
15. Zhang, R., Lee, H., Radev, D.: Dependency sensitive convolutional neural networks for modeling sentences and documents. arXiv preprint arXiv:1611.02361 (2016)
16. Hsu, S.T., Moon, C., Jones, P., Samatova, N.F.: A hybrid cnn-rnn alignment model for phrase-aware sentence classification. EACL 2017 (2017) 443
17. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A.J., Hovy, E.H.: Hierarchical attention networks for document classification. In: HLT-NAACL. (2016) 1480–1489
18. Zhao, Z., Wu, Y.: Attention-based convolutional neural networks for sentence classification. In: INTERSPEECH. (2016) 705–709
19. Zhou, X., Wan, X., Xiao, J.: Attention-based lstm network for cross-lingual sentiment classification. In: EMNLP. (2016) 247–256
20. Conneau, A., Schwenk, H., Barrault, L., Lecun, Y.: Very deep convolutional networks for text classification. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. Volume 1. (2017) 1107–1116
21. Gan, Z., Pu, Y., Henao, R., Li, C., He, X., Carin, L.: Learning generic sentence representations using convolutional neural networks. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. (2017) 2390–2400
22. McCann, B., Bradbury, J., Xiong, C., Socher, R.: Learned in translation: Contextualized word vectors. In: Advances in Neural Information Processing Systems. (2017) 6297–6308
23. Zhang, Y., Lease, M., Wallace, B.C.: Active discriminative text representation learning. In: AAAI. (2017) 3386–3392
24. Li, W., Mak, B.: Derivation of document vectors from adaptation of lstm language model. EACL 2017 (2017) 456
25. Zhao, R., Mao, K., Zhao, R., Mao, K.: Topic-aware deep compositional models for sentence classification. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) **25** (2017) 248–260
26. Li, S., Zhao, Z., Liu, T., Hu, R., Du, X.: Initializing convolutional filters with semantic features for text classification. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. (2017) 1885–1890

27. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification. In: Advances in neural information processing systems. (2015) 649–657
28. Zhang, X., LeCun, Y.: Text understanding from scratch. arXiv preprint arXiv:1502.01710 (2015)
29. Wehrmann, J., Becker, W., Cagnini, H.E., Barros, R.C.: A character-based convolutional neural network for language-agnostic twitter sentiment analysis. In: Neural Networks (IJCNN), 2017 International Joint Conference on, IEEE (2017) 2384–2391
30. Becker, W., Wehrmann, J., Cagnini, H., Barros, R.: An efficient deep neural architecture for multilingual sentiment analysis in twitter. (2017)
31. Yin, W., Schütze, H.: Multichannel variable-size convolution for sentence classification. arXiv preprint arXiv:1603.04513 (2016)
32. Zhang, Y., Roller, S., Wallace, B.: Mgnc-cnn: A simple approach to exploiting multiple word embeddings for sentence classification. arXiv preprint arXiv:1603.00968 (2016)
33. Deriu, J., Lucchi, A., De Luca, V., Severyn, A., Müller, S., Cieliebak, M., Hofmann, T., Jaggi, M.: Leveraging large amounts of weakly supervised data for multi-language sentiment classification. In: Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee (2017) 1045–1052
34. Singhal, P., Bhattacharyya, P.: Borrow a little from your rich cousin: Using embeddings and polarities of english words for multilingual sentiment classification. In: COLING. (2016) 3053–3062
35. Moreo Fernández, A., Esuli, A., Sebastiani, F.: Lightweight random indexing for polylingual text classification. Journal of Artificial Intelligence Research **57** (2016) 151–185
36. Pinto, D., Civera, J., Barrón-Cedeno, A., Juan, A., Rosso, P.: A statistical approach to crosslingual natural language tasks. Journal of Algorithms **64** (2009) 51–60
37. Lu, B., Tan, C., Cardie, C., Tsou, B.K.: Joint bilingual sentiment classification with unlabeled parallel corpora. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1, Association for Computational Linguistics (2011) 320–330
38. Song, Y., Mayhew, S., Roth, D.: Cross-lingual dataless classification for languages with small wikipedia presence. arXiv preprint arXiv:1611.04122 (2016)
39. Wan, X.: Co-training for cross-lingual sentiment classification. In: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-volume 1, Association for Computational Linguistics (2009) 235–243
40. Zhou, X., Wan, X., Xiao, J.: Cross-lingual sentiment classification with bilingual document representation learning. In: ACL (1). (2016)
41. Chen, X., Sun, Y., Athiwaratkun, B., Cardie, C., Weinberger, K.: Adversarial deep averaging networks for cross-lingual sentiment classification. arXiv preprint arXiv:1606.01614 (2016)
42. Mukku, S.S., Choudhary, N., Mamidi, R.: Enhanced sentiment classification of telugu text using ml techniques. In: SAAIP@ IJCAI. (2016) 29–34
43. Li, X., Roth, D.: Learning question classifiers. In: Proceedings of the 19th international conference on Computational linguistics-Volume 1, Association for Computational Linguistics (2002) 1–7
44. Pang, B., Lee, L.: Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In: Proceedings of the ACL. (2005)

45. Mukku, S.S., Mamidi, R.: Actsa: Annotated corpus for telugu sentiment analysis. In: Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems. (2017) 54–58
46. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9** (1997) 1735–1780
47. Chollet, F., et al.: Keras. `https://github.com/keras-team/keras` (2015)
48. Graves, A., Jaitly, N., Mohamed, A.r.: Hybrid speech recognition with deep bidirectional lstm. In: Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on, IEEE (2013) 273–278