Classification Of Spanish Election Tweets (COSET) with neural networks

Luis Cebrián Chuliá[1] and Sergio Ferrer Sánchez[1]

[1]Universitat Politècnica de València
{luicebch, serfers2}@inf.upv.es

Abstract. Obtaining information from tweets has become a field of interest in recent years due to its power to provide information about the insights of the users when any relevant event occurs. This is useful for companies and political parties that take advantage of this information in order to plan their next actions or to know whether or not their current actions are being received well by their public. In this work we show our approach for addressing COSET shared task (a tweet classification problem) using neural networks trained only with the data provided. With this approach we achieve the third position in the competition.

Keywords: Neural Networks, MLP, RNN

1 Introduction

In the last ten years, social media, such as Twitter or Facebook, has experienced a rapid growth that has changed the perspective of the socialization and public communication [1]. Social media are being used as a data source to analyze personal information about their users, such as opinions, likes or even political leaning since their publications provide a huge amount of data about themselves.

In Twitter, political conversation is a common topic and its influence increases when a General Election comes close since users speak more often about politicians or policies to criticize or praise them. This gives a lot of information related to how politicians or political groups are seen by citizenship. Also, detecting the most discussed topics is a key aspect in order to know people's concerns. As a result, this information can be used strategically during a political campaign to focus on those topics that citizens have doubts or want to address immediately.

In this context, COSET [4] shared task is focused on the classification of a dataset of tweets gathered during the 2015 Spanish General Election into five different classes, depending on the political topic discussed. To address this task, we will build a system based on neural networks that will tackle this problem.

2 Task Description

In this section we will be explaining the COSET shared task, part of the IberEval 2017 workshop. This shared task aims to classify tweets depending on the po-

litical topic discussed. The corpus has been extracted from conversations during the General Elections of 2015 in Spain. The dataset consists of 2242 tweets for training, 250 for development and 624 for test. As a final note, the tweets are presented in Spanish.

The different topics in which we have to classify the different tweets are:

- **Political issues.** Related to the most abstract electoral confrontation.
- **Policy issues.** Tweets about sectorial policies.
- **Campaign issues.** Related to the evolution of the campaign.
- **Personal issues.** Related to the personal life and activities of the candidates.
- **Other issues.**

The distribution of tweets among the above topics is very imbalanced being the largest class (Policy issues) of a size of approx. 1100 tweets and the smallest (Personal issues) approx. 200 tweets in size.

3 Experimental design

In this section we will cover all the experimentation carried out. First we will focus on feature extraction and text representation. Then we will address the classification problem by describing the architectures used.

3.1 Text preprocessing

In order to transform the text into a continuous representation we apply some preprocessing to clean as much as possible the data at hand. We have applied the Tweet tokenizer from the NLTK python library [2] with the flags for removing usernames, limiting the number of consecutive equal characters to three and applying lower case to capital letters. Links and emails have been left untouched.

Although usernames could seem useful, we decided to remove them because they did not contribute to improve the overall performance of the system and they also increased the size of the vocabulary by as much as 20%.

3.2 Text representation

We have approached the tweet classification problem in a variety of ways, most of them with regards to the text representation issue. Here we will only name the various alternatives we have tested. Results will be presented in next section. As our first approach we have tried representing the tweets with a fixed length representation:

- **Bag of n-grams**. To preserve information of both the vocabulary and the word order in the tweet.
- **Tf-Idf**. To account for frequent and important words.

– **N-gram of characters**. Similar to the previous approach but at character level.

Considering the great performance Recurrent Neural Networks (RNN) are having while processing sequences[7,9], we have also tried with a non-fixed length representation with:

– **Word embedding**. Words in tweets are represented as vectors, being those vectors a continuous representation of words.
– **Character embedding**. Similar to the word embedding but at a character level.

3.3 Architecture for fixed-length representation: Multilayer perceptron

When dealing with fixed-length representation a multilayer perceptron (MLP) is used to tackle the classification problem. Due to the size of the training dataset and the size of each sample, the expressiveness of this model is more than enough to learn the representations explained before.

The first few experiments were carried with a two-layer perceptron with 512 units in each layer. This model suffered from overfitting very quickly (99.5% accuracy in training in 5 epochs), that is why we reduced the layer's size to 64 units each. With this configuration the overfitting still occurred but it was manageable with noise.

Moreover, the best 4 models (trained with different initializations but identical configuration of parameters) were combined forming an ensemble to allow some collaboration among them in order to achieve better classification performance.

The combination of models has been implemented in a very straightforward way. The output of the last layer of all models is added and averaged by the number of models. The resulting vector is the one used to select the given class.

The final architecture (used to submit the best results) is a two-layer perceptron with 64 units in each hidden layer. The activation function used on the hidden layers is ReLU while the one used on the output layer is Softmax. Gaussian noise is used on the hidden layers with 0.5 standard deviation. The model was trained during 13 epochs feeding the model with batches of size 128. No class weighting was used to compensate imbalanced classes.

The model is fed with tweets represented as bag of n-grams with the n-gram size up to 3. This results in a size of vocabulary of $\sim$47k. The training of this model takes no more than 30 seconds on cpu.

3.4 Architecture for variable-length representation: Recurrent neural network

In the case of variable-length representation we used recurrent neural networks (RNN). With this model, the tweets are represented as a sequence of vectors of

real numbers. These vectors are fed sequentially to the RNN which will obtain a fixed length vector containing all the information extracted from the tweet. Then a multilayer perceptron will take that as input and perform the classification task as stated in Section 3.3. We use 128 as the embedding size and 64 as the hidden state of the recurrent unit. The multilayer perceptron has 64 units on its hidden layer.

In order to transform the text into a sequence of vectors of real numbers we must project our vocabulary into an embedding layer which will hold the continuous representation of every word (or character, depending on the representation) on the vocabulary. This continuous representation will be learned during the training process (no pretrained word embeddings used).

Moreover, additional tests have been made with a bidirectional RNN [8] (analyzes the tweet from the start and from the end) and with the use of Gated Recurrent Units (GRU, [3]) and Long-Short Term Memory (LSTM, [6]).

4 Results

In Table 1 are compiled all the results obtained by training different models with different representations of the data and different configurations. By looking at the table we can see that the best results are those obtained with the multilayer perceptron with bag of words representation. We can also see that other representations like tf-idf or other sizes of n-grams obtain similar results although with a slightly lower score. However, with better fine-tuning we were able to get better models that achieved several points above the ones presented in the results with the bag of 3-grams representation. For example, the model that generated the submission file achieved 62.4 f1-measure on the development set.

Ensembles are used to try to obtain better results by combining the same model with different initialization. This has been proven to yield better results [5,10]. However we find that in our case we do not gain much. Results obtained with ensembles are similar to the ones obtained with each model of the given ensemble.

Special attention has been paid to select which models were used to be part of the ensemble because the F-measure obtained was greatly influenced with the performance obtained in one of the classes (usually, the one with fewer samples). If this had not been taken into account, the results obtained would have been much worse.

In the case of RNNs they have performed very poorly compared to the MLP probably due to the short length of the tweets, noisy behavior and low number of samples.

Finally, we only see a slight improvement with the character-wise representation when we use tf-idf bag of n-grams. All the others configurations performed worse than its word-wise variants.

Table 1: Results obtained (measured with F-measure) with different models and representations. BoW stands for bag of words (unigrams). BiRNN stands for bidirectional RNN. Models marked with *ensemble* are a combination of 4 models initialized differently. The model used to submit the results is highlighted.

Model (word level)	F1-measure	Model (char level)	F1-measure
MLP-BoW	58.68 ± 0.51	MLP-2grams	46.79 ± 0.71
MLP-2grams	57.38 ± 0.34	MLP-3grams	53.55 ± 1.16
MLP-3grams	**56.55 ± 0.52**	MLP-4grams	55.25 ± 0.94
MLP-3grams-ensemble	57.10 ± 0.63	MLP-4grams-ensemble	56.12 ± 0.78
MLP-Tf-idf-BoW	57.77 ± 0.30	MLP-Tf-idf-2grams	50.84 ± 0.29
MLP-Tf-idf-2grams	56.04 ± 0.24	MLP-Tf-idf-3grams	56.63 ± 0.47
MLP-Tf-idf-3grams	55.77 ± 0.24	MLP-Tf-idf-4grams	58.19 ± 0.33
MLP-Tf-idf-ensemble	55.31 ± 0.23	MLP-Tf-idf-ensemble	57.31 ± 0.55
RNN-GRU	46.92 ± 0.98	RNN-GRU	29.85 ± 1.12
RNN-LSTM	48.20 ± 0.81	RNN-LSTM	31.19 ± 1.23
RNN-ensemble	45.32 ± 0.76	RNN-ensemble	30.24 ± 0.94
BiRNN-LSTM	47.91 ± 0.86	BiRNN-LSTM	30.94 ± 0.81
BiRNN-GRU	46.89 ± 0.71	BiRNN-GRU	28.49 ± 1.11

5 Conclusion

Various ways for classifying tweets have been tested in order to find which model suits the task the best. From a simple multilayer perceptron to RNN with gated units. Models trained with the first architecture (MLP) yielded the best results while the ones obtained with RNN were significantly lower.

We think that this is due to the small size of the dataset with just a few thousands of samples. This has a major drawback: great overfitting (and then, poor generalization). The expressiveness of the MLP is enough to achieve an acceptable performance with the data at hand while being in control of the overfitting. However, although RNNs have a greater expressiveness, they also need much more data, especially to learn the word embedding, in order to accomplish good generalization. This makes RNN much more suited to tasks where the number of samples is big enough to learn its underlying characteristics.

The MLP architecture with the bag of 3-grams representation used in our experiments achieved the 3rd best result of all participants in COSET shared task.

References

1. Batool, R., Khan, W., Hussain, M., Maqbool, J., Afzal, M., Lee, S.: Towards personalized health profiling in social network. Proceedings of the 6th International Conference on New Trends in Information Science, Service Science and Data Mining (2012)
2. Bird, S., Klein, E., Loper, E.: Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc." (2009)

3. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555 (2014)
4. Giménez, M., Baviera, T., Llorca, G., Gámir, J., Calvo, D., Rosso, P., Rangel, F.: Overview of the 1st classification of spanish election tweets task at ibereval 2017. In: Proceedings of the Second Workshop on Evaluation of Human Language Technologies for Iberian Languages (IberEval 2017), Murcia, Spain, September 19, CEUR Workshop Proceedings. CEUR-WS.org, 2017
5. Hansen, L.K., Salamon, P.: Neural network ensembles. IEEE transactions on pattern analysis and machine intelligence 12(10), 993–1001 (1990)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9(8), 1735–1780 (1997)
7. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech. vol. 2, p. 3 (2010)
8. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing 45(11), 2673–2681 (1997)
9. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in neural information processing systems. pp. 3104–3112 (2014)
10. West, D., Dellana, S., Qian, J.: Neural network ensemble strategies for financial decision applications. Computers & operations research 32(10), 2543–2559 (2005)