LIA at SemEval-2017 Task 4: An Ensemble of Neural Networks for Sentiment Classification

Mickael Rouvier
LIA
University of Avignon
Avignon, France
mickael.rouvier@univ-avignon.fr

Abstract

This paper describes the system developed at LIA for the SemEval-2017 evaluation campaign. The goal of Task 4.A was to identify sentiment polarity in tweets. The system is an ensemble of Deep Neural Network (DNN) models: Convolutional Neural Network (CNN) and Recurrent Neural Network Long Short-Term Memory (RNN-LSTM). We initialize the input representation of DNN with different sets of embeddings trained on large datasets. The ensemble of DNNs are combined using a score-level fusion approach. The system ranked 2^{nd} at SemEval-2017 and obtained an average recall of 67.6%.

1 Introduction

This paper describes the system developed at LIA for the SemEval-2017 sentiment analysis evaluation task 4 (Rosenthal et al., 2017).

We have participated in Subtask A: sentiment analysis at the message level in English. It consists in determining the message polarity of each tweet in the test set. The sentiment polarity classification task is set as a three-class problem: positive, negative and neutral.

The sentiment analysis task is often modeled as a classification problem which relies on features extracted from the text in order to feed a classifier. Recent work has shown that Deep Neural Networks (DNN) using word representations as input are well suited for sentence classification problems and have been shown to produce state-of-the-art results for sentiment polarity classification (Tang et al., 2014a; Severyn and Moschitti, 2015). Two different types of DNN models are used: Convolutional Neural Network (CNN) and Recurrent Neural Network with Long Short-Term Memory units (RNN-LSTM). Pretrained word embeddings are used to initialize the word representations, which are then taken as input of a text.

Our approach consists in learning classifiers for four types of embeddings, based on the CNN and RNN-LSTM architectures. Each set of word embeddings models the tweet according to a different point of view. A final fusion step is applied.

Our contributions are as follows:

- We propose to apply a teacher-student appraoch for training the DNN models.
- We propose a new way to capture polarity in word embeddings.
- The source code of our system, the models trained for the evaluation, as well as the corpus collected for creating word embeddings, are all made available to the community in hope of helping future research [1].

The paper is structured as follows. Section 2 presents a quick overview of the system architecture, which is then detailed in sections 3 and 4, along with the various word embeddings and other features used in our system. Results and discussion appear in Section 5.

2 Overview of the approach

The system was developed as a two-level architecture. Given a tweet, the first level extracts input representations based various word embeddings. These

[1] http://gitlia.univ-avignon.fr/rouvierm/semeval-2017-sentiment-analysis

Proceedings of the 11th International Workshop on Semantic Evaluations (SemEval-2017), pages 760–765,
Vancouver, Canada, August 3 - 4, 2017.

embeddings are fed to a DNN model (CNN and RNN-LSTM). Four different sets of word embeddings are used: one lexical embedding and three different sentiment embeddings.

The second level inputs the concatenation of the scores obtained each input representation and Deep Neural Network (DNN and RNN-LSTM) from the first level. This representation is fed to a Multi-Layer Perceptron (MLP) which was trained to predict polarity.

3 Deep Neural Networks

3.1 Convolutional Neural Networks

CNNs represent one of the most used Deep Neural Network models in computer vision (LeCun and Bengio, 1995). Recent work has shown that CNNs are also well suited for sentence classification problems and can produce state-of-the-art results (Tang et al., 2014a; Severyn and Moschitti, 2015). The difference between CNNs applied to computer vision and their equivalent in NLP lies in the input dimensionality and format. In computer vision, inputs are usually single-channel (eg. grayscale) or multi-channel (eg. RGB) 2D or 3D matrices, usually of constant dimension.

In sentence classification, each input consists of a sequence of words of variable length. Each word w is represented with a n-dimensional vector (word embedding) e_w of constant size. All the word representations are then concatenated in their respective order and padded with zero-vectors.

The parameters of our model were chosen so as to maximize performance on the development set: the width of the convolution filters is set to 5 and the number of convolutional feature maps is 300. We use ReLU activation functions and a simple max-pooling. The fully-connected hidden layer is of size 512.

For this layer, a standard dropout of 0.4 is used (40 % of the neurons are disabled at each iteration). The back-propagation algorithm used for training is Adadelta. In our experiments we observed that the weight initialization of the convolution layer can lead to a high variation in terms of performance. Therefore, we trained 20 models and selected the one that obtained the best results on the development corpus.

3.2 Reccurent Neural Network with Long Short Term Memory

RNNs are popular models that have shown great promise in many Natural Language Processing (NLP) tasks. The main differentiating feature of RNNs is that the model take into account the ordering of words in the text as opposed to CNNs which take only a limited, small context window.

In a traditional neural network we assume that all inputs (and outputs) are independent of each other. RNNs can take into account the input but also what they perceived one step back in time. Hence recurrent networks have two sources of input, the present and the recent past, which combine to determine how to respond to new data, much as we do in life.

The parameters of our model were also chosen so as to maximize performance on the development set: the hidden layer is of size 128, a standard dropout of 0.2 is used. The back-propagation algorithm used for training is Adadelta.

3.3 Word embeddings

Word embeddings are an approach for distributional semantics which represents words as vectors of real numbers. Such a representation has useful clustering properties, since it groups together words that are semantically and syntactically similar (Mikolov et al., 2013). For example, the word "coffee" and "tea" will be very close in the created space. The goal is to use these features as input to a DNN classifier. However, with the sentiment analysis task in mind, typical word embeddings extracted from lexical context might not be the most accurate because antonyms tend to be placed at the same location in the created space.

This year, in SemEval-2017 we explored different approaches to integrate the sentiment polarity of the words. Four representations were explored:

Lexical embeddings: these embeddings are obtained with the classical skipgram model from (Mikolov et al., 2013). The representation is created by using the hidden layer of a linear neural network to predict a context window from a central word. For a given context $w_{i-2} \dots w_{i+2}$, the input to the model is w_i, and the output could be w_{i-2}, w_{i-1}, w_{i+1}, w_{i+2}. This method typically extracts a representation which covers both syntax and semantics, to some extent.

Sentiment embeddings (multitask-learning): One

of the problems with the basic skipgram approach (lexical embeddings) is that the model ignores the sentiment polarity of the words. As a result, words with opposite polarity, such as "good" and "bad", are mapped into close vectors. In (Tang et al., 2014b), the authors propose to tackle this problem so that sentiment information is encoded in the continuous representation of words. They propose to create a neural network that predicts two tasks: the context of the word and the sentiment label of the whole sentence. Since it is expensive to manually label sentences with their polarity, the authors propose to use tweets that contain emoticons and rely on the polarity conveyed by the emoticons to label the sentences. Since they report that best performance is obtained by weighting both tasks equivalently, the model is the same as for lexical embeddings, except that the predicted context is formed of (word, sentiment) couples. For example, if s is the polarity of the sentence where the context $w_{i-2} \dots w_{i+2}$ is extracted, the model gets w_i as input and has to predict $(w_{i-2}, s), (w_{i-1}, s), (w_{i+1}, s), (w_{i+2}, s)$.

Sentiment embeddings (distant-supervision): The distant-supervision is another solution to integrating sentiment polarity in words. A DNN (CNN or RNN-LSTM) is trained on massive distant-supervised tweets selected by positive and negative emoticons. The positive and negative emoticons are used as supervised labels. During training, the DNN will automatically refine the word embeddings in order to capture the sentiment polarity in words. The refined word embeddings can be used as a new representation.

Sentiment embeddings (negative-sampling): The negative-sampling approach is an efficient way of computing softmax. In order to deal with the difficulty of having too many output vectors that need to be updated, the main idea of negative sampling is to update not all the words but only a few words as negative samples (hence "negative sampling"). Instead of selecting random words, as is usual for this technique, we chose to select words with opposite polarities. For example, for the word "good" we select the words "bad", "terrific", etc. for negative sampling.

3.4 Extension of the DNN model

The DNN model relies on word embeddings as word representation. Unfortunately these models can only capture information at the word level. We propose to extract some sentence-level information and to inject this information into the model. In order to incorporate this source of information into the system, a set of sentence-level features are concatenated with the last hidden layer in the model.

The following features are extracted at the sentence level:

- **Lexicons**: frequency of lemmas that are matched in MPQA (Wiebe et al., 2005), Opinion Lexicon (Hu and Liu, 2004) and NRC Emotion lexicon (Mohammad and Turney, 2013).
- **Emoticons**: number of emoticons that are grouped in positive, negative and neutral categories.
- **All-caps**: number of words in all-caps.
- **Elongated units**: number of words in which characters are repeated more than twice (for example: looooool).
- **Punctuation**: number of contiguous sequences of several periods, exclamation marks and question marks.

3.5 Mimic model

The teacher-student approach consists in training a state-of-the-art model (teacher model), and then training a new model (student model) to mimic the teacher model. The mimic model (student model) is not trained on the original labels, but it is trained to learn targets predicted by the teacher model. Remarkably, a mimic model trained on targets predicted by the teacher model can be more accurate than teacher model trained on the original labels. There are a variety of reasons why this can happen:

- If some labels have errors, the teacher model may eliminate some of these errors thus making it learning easier for the student.
- Learning from the original, hard 0/1 labels can be more difficult than learning from a teacher's conditional probabilities; but the mimic model sees non-zero targets for most outputs on most training cases, and the teacher can spread uncertainty over multiple outputs for difficult cases. The uncertainty from the teacher model is more informative to the student model than the original 0/1 labels.
- The mimic model can be seen as a form of regularization that helps prevent overfitting the model.

Corpus	Positive	Negative	Neutral	Total
SemEval_{13-15}	9.316	3.443	9.067	21.826
SemEval_{16}	7.059	3.231	10.342	20.632

Table 1: Statistics of the successfully downloaded part of the SemEval 2017 Twitter sentiment classification dataset.

System	Rec^{Pos}	Rec^{Neu}	Rec^{Neg}	Avg-Rec
Fusion	64.6	57.7	80.4	67.6

Table 2: Overall performance of the LIA sentiment analysis systems. Rec^{Pos}, Rec^{Neu} and Rec^{Neg} are respectively the recall on positive, neutral and negative classe. Avg-Rec is the macro-averaged recall calculated over the three categories.

4 Fusion system

The outputs from all Deep Neural Networks are concatenated to form a single feature vector. This vector is then fed into a Multi-Layer Perceptron (MLP) that is trained to predict the polarity. The MLP contains one hidden layer of 128 neurons and the activation function used is $tanh$.

5 Experiments

5.1 Corpus

We use the training corpus from Twitter'13 to 17 for training the various parts of the architecture. We split the corpus into two parts. The train, development and test corpora given in SemEval'13, 14, 15 form the first part, referred to as SemEval_{13-15}. The test corpus given in SemEval'16 forms part 2, referred to as SemEval_{16}. We perform 2-fold cross-validation, where one part is used as the training corpus and the other one is used as the development corpus. Note that we were unable to download all the training and development data because some tweets were deleted or not available due to modified authorization status. The sizes of the datasets are summarized in Table 1.

5.2 Word embedding training

To train the word embeddings, we have created a unannotated corpus of sentiment-bearing tweets in English. These tweets were recovered on the Twitter platform by searching for emotion keywords (from the sentiment lexicons) and unigrams, bigrams and trigrams extracted from the SemEval training corpus. This corpus consists of about 90 million tweets. A sub-corpus of about 20 million tweets containing at least one emoticon is used for training the sentiment embeddings. Both corpora are now publicly available [2].

In our experiments, lexical embeddings and part-of-speech embeddings are estimated using the word2vec toolkit (Mikolov et al., 2013). Sentiment embeddings are estimated using word2vecf. This toolkit allows to replace linear bag-of-word contexts with arbitrary features. The embeddings are trained using the skipgram approach with a window of size 3 and 5 iterations. The dimension of the embeddings is set to 100. Part-of-speech tagging is performed with Tweet NLP (Owoputi et al., 2013; Gimpel et al., 2011).

[2] http://gitlia.univ-avignon.fr/rouvierm/semeval-2017-sentiment-analysis

5.3 Results

Overall performance: The evaluation metric used in the competition is the macro-averaged recall calculated over the three categories. Table 3 presents the overall performance of each of the systems used for the first level. We observe that the best first-level system is the CNN Mimic model using sentiment embedding (negative-sampling). The system obtained an average-recall of 66.91%. Concerning the word embedding, in generally sentiment embedding (negative-sampling) obtains for each DNN model the best results. Concerning the DNN models, the CNN approach provide better results than the RNN-LSTM models.

Impact of fusion: Table 2 presents the results obtained by the fusion system. It achieved the second rank on the Twitter 2017 data among 39 teams.

6 Conclusions

This paper describes the LIA participation in SemEval 2017. Our approach consists in running an ensemble of neural networks (CNN and RNN-LSTM) over different types of embeddings. A final fusion step is applied, based on concatenating the scores given by the neural networks and training a deep neural network for the fusion. The resulting system ranked 2^{nd} at the SemEval-2017 evaluation campaign.

Acknowledgments

This project was financed by the project CHISTERA CALL - ANR: Access Multilingual Information opinionS (AMIS), (France - Europe).

References

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 42–47. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *SIGKDD*, pages 168–177.

Yann LeCun and Yoshua Bengio. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10).

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Saif M Mohammad and Peter D Turney. 2013. Nrc emotion lexicon. Technical report, NRC Technical Report.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics.

Sara Rosenthal, Noura Farra, and Preslav Nakov. 2017. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation*, SemEval '17, Vancouver, Canada, August. Association for Computational Linguistics.

Aliaksei Severyn and Alessandro Moschitti. 2015. Unitn: Training deep convolutional neural network for twitter sentiment classification. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015), Association for Computational Linguistics, Denver, Colorado*, pages 464–469.

Duyu Tang, Furu Wei, Bing Qin, Ting Liu, and Ming Zhou. 2014a. Coooolll: A deep learning system for twitter sentiment classification. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 208–212.

Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014b. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL (1)*, pages 1555–1565.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210.

Appendix

System	Architecture	Word embeddings	Training corpus	Rec^{Pos}	Rec^{Neu}	Rec^{Neg}	Avg-Rec
1	CNN	lexical	SemEval$_{13-15}$	63.5	53.9	80.3	65.9
2	CNN	sentiment (multi-task learning)	SemEval$_{13-15}$	66.4	50.0	82.5	66.3
3	CNN	sentiment (distant-supervision)	SemEval$_{13-15}$	65.5	49.8	83.9	66.4
4	CNN	sentiment (negative-sampling)	SemEval$_{13-15}$	65.6	51.5	81.4	66.2
5	RNN-LSTM	lexical	SemEval$_{13-15}$	56.0	58.6	76.9	63.9
6	RNN-LSTM	sentiment (multi-task learning)	SemEval$_{13-15}$	62.4	52.4	78.0	64.3
7	RNN-LSTM	sentiment (distant-supervision)	SemEval$_{13-15}$	59.5	60.0	68.6	62.7
8	RNN-LSTM	sentiment (negative-sampling)	SemEval$_{13-15}$	61.5	60.9	72.2	64.9
9	CNN Mimic	lexical	SemEval$_{13-15}$	66.3	53.7	80.0	66.7
10	CNN Mimic	sentiment (multi-task learning)	SemEval$_{13-15}$	64.6	52.8	81.7	66.4
11	CNN Mimic	sentiment (distant-supervision)	SemEval$_{13-15}$	65.0	51.2	83.8	66.7
12	CNN Mimic	sentiment (negative-sampling)	SemEval$_{13-15}$	70.6	48.0	82.3	66.9
13	CNN	lexical	SemEval$_{16}$	57.0	58.6	80.9	65.5
14	CNN	sentiment (multi-task learning)	SemEval$_{16}$	53.2	53.5	84.2	63.6
15	CNN	sentiment (distant-supervision)	SemEval$_{16}$	56.3	58.3	82.1	65.5
16	CNN	sentiment (negative-sampling)	SemEval$_{16}$	60.3	45.8	88.7	64.9
17	RNN-LSTM	lexical	SemEval$_{16}$	55.8	58.4	78.2	61.4
18	RNN-LSTM	sentiment (multi-task learning)	SemEval$_{16}$	56.1	55.5	81.3	64.3
19	RNN-LSTM	sentiment (distant-supervision)	SemEval$_{16}$	52.6	48.0	83.7	61.5
20	RNN-LSTM	sentiment (negative-sampling)	SemEval$_{16}$	61.7	62.7	71.0	65.1
21	CNN Mimic	lexical	SemEval$_{16}$	57.5	53.3	84.1	65.0
22	CNN Mimic	sentiment (multi-task learning)	SemEval$_{16}$	59.8	53.9	82.9	65.5
23	CNN Mimic	sentiment (distant-supervision)	SemEval$_{16}$	58.9	53.8	83.3	65.3
24	CNN Mimic	sentiment (negative-sampling)	SemEval$_{16}$	62.1	52.6	83.8	66.2

Table 3: Overall performance of the DNN models using different word embeddings and training corpus. Rec^{Pos}, Rec^{Neu} and Rec^{Neg} are respectively the recall on positive, neutral and negative classe. Avg-Rec is the macro-averaged recall calculated over the three categories.