ECNU at SemEval-2018 Task 2: Leverage Traditional NLP Features and Neural Networks Methods to Address Twitter Emoji Prediction Task

Xingwu Lu[1], **Xin Mao**[1], **Man Lan**[1,2*], **Yuanbin Wu**[1,2]
[1]Department of Computer Science and Technology,
East China Normal University, Shanghai, P.R.China
[2]Shanghai Key Laboratory of Multidimensional Information Processing
{51174506023, 10131530334}@stu.ecnu.edu.cn
{mlan, ybwu}@cs.ecnu.edu.cn

Abstract

This paper describes our submissions to Task 2 in SemEval 2018, i.e., Multilingual Emoji Prediction. We first investigate several traditional Natural Language Processing (NLP) features, and then design several deep learning models. For subtask 1: Emoji Prediction in English, we combine two different methods to represent tweet, i.e., supervised model using traditional features and deep learning model. For subtask 2: Emoji Prediction in Spanish, we only use deep learning model.

1 Introduction

Visual icons play a crucial role in providing information about the extra level of social media information. SemEval 2018 shared task for researchers to predict, given a tweet in English or Spanish, its most likely associated emoji (Barbieri et al., 2018, 2017) (Task 2, Multilingual Emoji Prediction), which is organized into two optional subtask (subtask 1 and subtask 2) respectively in English and Spanish.

For subtask 1, we adopt a combination model to predict emojis, which consists of traditional Natural Language Processing (NLP) methods and deep learning methods. The results returned by the classifier with traditional NLP features, by the neural network model and by the combination model are voted to get the final result. For subtask 2, we only use deep learning model.

2 System Description

For subtask 1, we explore three different methods i.e., using traditional NLP features to learn a supervised machine learning-based classifier, learning a deep learning model to make prediction and combine features captured by neural networks with traditional NLP features to train a supervised machine learning-based classifier. For subtask 2, we simply implement deep learning method to make prediction.

2.1 Traditional NLP Features

In this task, we extract the following three types of features to capture effective information from the given tweets, i.e., linguistic features, sentiment lexicon features and tweet specific features.

2.1.1 Linguistic Features

- *N-grams:* We extract 3 types of Bag-of-Words features as *N-grams* features, where N = 1,2,3 (i.e., *unigram*, *bigram*, and *trigram* features).

- *POS:* Generally, the sentences carrying subjective emotions are inclined to contain more adjectives and adverbs while the sentences without sentiment orientation would contain more nouns. Thus, we extract POS tag from the sentence as features with the Bag-of-Words form.

- *Correlation Degree:* For each word appear in training data, the ratio of the number of occurrences under each class and the total occurrences is counted as the correlation degree of the word to a certain class. When the feature is created, the sum of the correlation degree of words in tweet is counted as the correlation degree of the tweet to a certain class:

$$CorrDeg(s,l) = \sum_{t=1}^{|s|} \frac{O(w_t, c_l)}{\sum_i^N O(w_t, c_i)}$$

Where $|s|$ is the length of tweet and N is the number of classes, w_t means t^{th} word in tweet and c_i means i^{th} class, $O(w_t, c_i)$ denotes the number of tweets of c_i that contain w_t. The dimension of this feature is equal to the number of classes, value is correlation

Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018), pages 433–437
New Orleans, Louisiana, June 5–6, 2018.

degree of the tweet to each class, i.e., *CorrDeg(s,l)*.

2.1.2 Sentiment Lexicon Features (SentiLexi)

We also extract sentiment lexicon features (SentiLexi) to capture the sentiment information of the given sentence. Given a tweet, we first convert all words into lowercase. Then on each sentiment lexicon, we calculate the following six scores for one message: (1) the ratio of positive words to all words, (2) the ratio of negative words to all words, (3) the maximum sentiment score, (4) the minimum sentiment score, (5) the sum of sentiment scores, (6) the sentiment score of the last word in tweet. If the word does not exist in one sentiment lexicon, its corresponding score is set to 0. The following 8 sentiment lexicons are adopted in our systems: *Bing Liu lexicon*[1], *General Inquirer lexicon*[2], *IMDB*[3], *MPQA*[4], *NRC Emotion Sentiment Lexicon*[5], *AFINN*[6], *NRC Hashtag Sentiment Lexicon*[7], and *NRC Sentiment140 Lexicon*[8].

2.1.3 Tweet Specific Features

- *Punctuation:* Considering that users often use exclamation marks and question marks to express strongly surprised and questioned feelings, we extract 7-dimensions punctuation features by recording rules of punctuation marks in the tweets.
- *All-caps:* One binary feature is to check whether this tweet has words in uppercase.
- *Bag-of-Hashtags:* We construct a vocabulary of hashtags appearing in the training data and then adopt the bag-of-hashtags method for each tweet.

2.2 Deep Learning Modules

In addition to manually constructing features, we build deep neural models to capture the semantics of the text. Figure 1 shows the network structure of our model. The input of the network is a tweet, which is a sequence of words. The output of the network contains class elements.

[1]http://www.cs.uic.edu/liub/FBS/sentiment-analysis.html
[2]http://www.wjh.harvard.edu/inquirer/homecat.htm
[3]http://www.aclweb.org/anthology/S13-2067
[4]http://mpqa.cs.pitt.edu/
[5]http://www.saifmohammad.com/WebPages/lexicons.html
[6]http://www2.imm.dtu.dk/pubdb/views/publication details.php?id=6010
[7]http://www.umiacs.umd.edu/saif/WebDocs/NRC-Hashtag-Sentiment-Lexicon-v0.1.zip
[8]http://help.sentiment140.com/for-students/

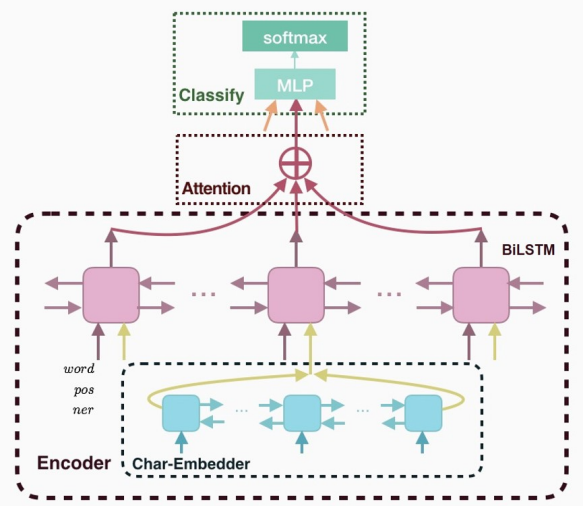

Figure 1: Deep learning model architecture.

2.2.1 Word-Level Representations

We use pre-trained word embedding concatenated with char embedding, POS embedding and NER embedding obtaining a final representation for each word type, which are learned together with the updates to the model.

- *Word Embedding:* Word embedding is a continuous-valued vector representation for each word, which can capture meaningful syntactic and semantic regularities. In this task, we use the 300-dimensional word vectors pre-trained on Twitter provided by SemEval task organizers, available in *SWM*[9]
- *Char Embedding:* We randomly initialize the representation of the character and compute character-based continuous-space vector embeddings of the words in tweets by bidirectional LSTM. The dimension of char embedding is 50.
- *POS Embedding:* We randomly initialize the representation of the POS tag in tweet with a vector size of 50.
- *NER Embedding:* We also randomly initialize the representation of the NER tag in tweet with a vector size of 50.

2.2.2 Sentence-Level Representations

- *Bi-Directional LSTM:* We apply a recurrent structure to capture contextual information as far as possible when learning word representations, to model the tweet with both of the

[9]https://github.com/fvancesco/acmmm2016

preceeding and following contexts, we apply a Bi-directional Long Short-term Memory Networks (BiLSTM, Graves et al. (2005)) architecture as shown in Figure 1.

- *Attention Mechanism:* Considering not all words contribute equally to the representation of the sentence meaning, we introduce attention mechanism (Bahdanau et al., 2014) to extract such words that are important to the meaning of the sentence and aggregate the representation of those informative words to form a sentence vector.

We first use BiLSTM and Attention Mechanism to obtain sentence-level representations and then concatenate it with several effective NLP features. At last, we use a Multi-layer Perceptron (MLP) and output the probability of emoji label based on a softmax function. The BiLSTM has a hidden size of 512. The MLP have 1 hidden layer of size 200 and *relu* non-linearity.

To learn model parameters, we minimize the KL-divergence between the outputs and gold labels. We adopt Adam (Kingma and Ba, 2014) as optimization method and set learning rate of 0.01.

3 Experimental Settings

3.1 Datasets

For training sets, the organizers provide only the list of tweet ID and a script for all participants to collect tweets. However, since not all tweets are available when downloading, participants may collect slightly different numbers of tweets for training sets. In addition, we find that the crawled training sets and the trial sets provided by the organizers have 37.26% overlap in English and 71.16% in Spanish. So we remove the duplicate data and combine train and trial sets to perform a 3-fold cross-validation. Table 1 shows the statistics of the tweets we collect in our experiments. In subtask 1, the number of class 0 is the largest, accounting for 22.28%, followed by class 1 and class 2, respectively, 10.37% and 10.20%, and the other 17 classes distribute between 2.46% and 5.51%. Subtask 2 has a similar data distribution.

3.2 Data Preprocessing

Firstly, we convert unicode encoding into corresponding characters, punctuation, emoticons. Then we use slangs[10] to transform the informal writing to regular forms, e.g., "*LOL*" replaced by "*laugh out loud*". And we recover the elongated words to their original forms, e.g., "*soooooo*" to "*so*". Finally, we implement tokenization, POS tagging, named entity recognizing(NER) with the aid of Stanford CoreNLP tools (Manning et al., 2014).

label \ Dataset	**Subtask 1**	**Subtask 2**
0	116693 (22.28%)	20495 (20.41%)
1	54313 (10.37%)	13688 (13.63%)
2	53432 (10.20%)	9342 (9.30%)
3	28855 (5.51%)	6859 (6.83%)
4	25778 (4.92%)	6535 (6.51%)
5	24475 (4.67%)	4765 (4.43%)
6	22301 (4.26%)	4444 (4.42%)
7	19236 (3.67%)	3868 (3.85%)
8	17908 (3.42%)	3687 (3.67%)
9	16996 (3.25%)	3544 (3.53%)
10	16755 (3.20%)	3399 (3.16%)
11	16052 (3.07%)	2943 (2.93%)
12	15167 (2.90%)	2826 (2.81%)
13	13650 (2.61%)	2727 (2.72%)
14	14136 (2.70%)	2629 (2.62%)
15	13963 (2.67%)	2564 (2.55%)
16	13702 (2.62%)	2618 (2.61%)
17	13474 (2.57%)	2551 (2.54%)
18	13867 (2.65%)	2552 (2.54%)
19	12900 (2.46%)	–
total	523653	100440

Table 1: The statistics of data sets in combination of training sets and trial sets which we used to perform a 3-fold cross-validation. The numbers in brackets are the percentages of different classes in each data set.

3.3 Learning Algorithm

Considering the large dimension of the features designed by traditional NLP methods, we use learning algorithms of Logistic Regression(LR) to build classification models, which is supplied in *Liblinear*[11].

3.4 Evaluation Metrics

The official evaluation measure is Macro F-score, which would inherently mean a better sensitivity to the use of emojis in general, rather than for instance overfitting a model to do well in the three or four most common emojis of the test data. Macro F-score can be defined as simply the average of the individual label-wise F-scores.

[10]https://github.com/haierlord/resource/blob/master/slangs

[11]https://www.csie.ntu.edu.tw/ cjlin/liblinear/

4 Experiments on Training Data

4.1 Comparison of NLP Features

Table 2 lists the comparison of different contributions made by different features on cross-validation with Logistic Regression algorithm. From the results in Table 2, we observe the following findings:

(1) The combination of Uigram, Bigram, Correlation Degree, POS and SentiLexi achieves the best performance (i.e., 34.63).

(2) Correlation Degree feature makes more contributes than other features, as it reflects the degree of relevance between tweets and emoji label.

(3) Bigram feature makes contribution and is more effective than unigram feature. The reason may be that bigram feature can capture more contextual information and word orders.

(4) SentiLexi feature also makes contribution, which indicates that SentiLexi features are beneficial not only in traditional sentiment analysis tasks, but also in predicting the emoji in tweet.

Features	$\mathbf{F_{macro}}$	change
Best Features	34.63	-
-Correlation Degree	30.59	-4.04
-Bigram	33.38	-1.25
-SentiLexi	33.63	-1.00
-POS	33.91	-0.72
-Uigram	34.22	-0.41

Table 2: Performance of different features on subtask 1. - means to exclude some features.

4.2 Comparison of Deep Learning Modules

Table 3 shows the results of different deep learning models described before. From Table 3, we observe the findings as follows:

(1) We explore the performance of three different deep learning model: Neural Bag-of-Words(NBOW, Iyyer et al. (2015)), Convolutional Neural Network (CNN, Collobert et al. (2011)) and Bi-directional Long Short-term Memory Networks (LSTM, Hochreiter and Schmidhuber (1997)). All models used only pre-trained word embedding to compare. Clearly, BiLSTM outperformed other models in this task, and our deep learning model is based on BiLSTM.

(2) POS embedding makes more contribution than other word-level representations. Since POS embedding can learn emotional tendencies, it is beneficial for tweet emojis prediction.

(3) The last two rows results shows that combine both SentiLexi and Punctuation features with sentence representations to train the deep learning model can make contribution.

Models	Subtask 1	Subtask 2
NBOW	23.73	18.67
CNN	23.64	18.91
BiLSTM	25.64	19.32
.+Char	25.66 (+0.02)	19.56
.+NER	26.57 (+0.91)	–
.+POS	30.55 (+3.98)	–
.+Attention	30.74 (+0.19)	–
.+Punctuation	32.10 (+1.36)	–
.+SentiLexi	32.59 (+0.49)	–

Table 3: Performance of different deep learning models on subtask 1 and subtask 2. .+ means to add current module to the previous model. The numbers in the brackets are the performance increments compared with the previous results.

4.3 Combination and Ensemble

For subtask 1, we also use the trained neural networks described in 4.2 to capture the features of tweets and combine it with traditional NLP features to train a Logistic Regression classifier, named *Combination Model*.

Table 4 shows the results of different methods. We find that combination model improved the performance and the ensemble of 3 methods achieve the best result. It suggests that the traditional NLP methods and the deep learning models are complementary to each other and their combination achieves the best performance.

Methods	$\mathbf{F_{macro}}$
Traditional NLP Features	34.63
Deep Learning Model	32.59
Combination Model	35.21
Ensemble Model	35.57

Table 4: Performance of different methods on subtask 1.

4.4 System Configuration

Based on above experimental analysis, the two system configurations on test data sets are listed as followings:

(1) subtask 1: Logistic Regression with best NLP feature sets is used as model 1. Deep learn-

ing model is used as model 2. Logistic Regression with NLP features and the feature captured by deep learning model is used as model 3. Ensemble of three models is used as final submission.

(2) subtask 2: Deep learning model with word embedding and char embedding is used as submission.

5 Results on Test Data

	Subtask 1	**Subtask 2**
Our system	33.35 (5)	16.41 (7)
Rank 1	35.99 (1)	22.36 (1)
Rank 2	35.36 (2)	18.73 (2)
Rank 3	34.02 (3)	18.18 (3)

Table 5: Performance of our systems and the top-ranked systems for two subtasks on test datasets. The numbers in the brackets are the official rankings.

Table 5 shows the results on test datasets. From Table 5, we find that our system achieves almost the same performance as the cross-validation. The low performance of this task illustrates the difficulty of the task itself, especially the Spanish task.

6 Conclusion

In this paper, we extract several effective traditional NLP features, design different deep learning models and build a model in combination of traditional NLP features and deep learning method together. The extensive experimental results show that this combination improves the performance.

For the future work, we consider to focus on developing a neural networks model to handle unbalanced data and improve the performance of confusing labels.

Acknowledgements

This work is is supported by the Science and Technology Commission of Shanghai Municipality Grant (No. 15ZR1410700) and the open project of Shanghai Key Laboratory of Trustworthy Computing (No.07dz22304201604).

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Francesco Barbieri, Miguel Ballesteros, and Horacio Saggion. 2017. Are emojis predictable? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 105–111. Association for Computational Linguistics.

Francesco Barbieri, Jose Camacho-Collados, Francesco Ronzano, Luis Espinosa-Anke, Miguel Ballesteros, Valerio Basile, Viviana Patti, and Horacio Saggion. 2018. SemEval-2018 Task 2: Multilingual Emoji Prediction. In *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, United States. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. 2005. Bidirectional lstm networks for improved phoneme classification and recognition. In *International Conference on Artificial Neural Networks*, pages 799–804. Springer.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of ACL 2015*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.