Tweet Emoji Prediction Using Hierarchical Model with Attention

Chuhan Wu
Tsinghua University
Beijing, 100084 China
wuch15@mails.tsinghua.edu.cn

Xing Xie
Microsoft Research Asia
Beijing, 100084 China
xing.xie@microsoft.com

Fangzhao Wu
Microsoft Research Asia
Beijing, 100084 China
wufangzhao@gmail.com

Sixing Wu
Tsinghua University
Beijing, 100084 China
wu-sx15@mails.tsinghua.edu.cn

Yongfeng Huang
Tsinghua University
Beijing, 100084 China
yfhuang@tsinghua.edu.cn

ACM.
UbiComp/ISWC'18 Adjunct,, October 8–12, 2018, Singapore, Singapore
ACM 978-1-4503-5966-5/18/10.
https://doi.org/10.1145/3267305.3274181

Abstract

With the development of social media, a huge number of users are attracted by social platforms such as Twitter. Emojis are widely used by social network users when posting messages. Therefore, it is important to mine the relationships between plain texts and emojis. In this paper, we present a neural approach to predict multiple emojis evoked by plain tweets. Our model contains three modules, i.e., a character encoder to learn representations of words from original characters using convolutional neural network (CNN), a sentence encoder to learn representations of sentences using a combination of long short-term memory (LSTM) network and CNN, a multi-label classification module to predict the emojis evoked by a tweet. Besides, attention mechanism is applied at word-level to select important contexts. Our approach is self-labeling and free from expensive and time-consuming manual annotation. Experiments on real-world datasets show that our model outperforms several automatic baselines as well as humans in this task.

Author Keywords

Emojis, Tweets, Neural Networks

ACM Classification Keywords

I.2.7 [Artificial Intelligence]: Natural Language Processing.

Introduction

In recent years, emojis such as 😂 become increasingly popular on social media platforms such as Twitter. Emojis are often combined with texts in web messages to convey different meanings and emotions [13, 3, 4]. It is an important task to mine the relationships between emojis and text-based messages. Analyzing the relationships between emojis and text-based messages is important and has many potential applications, such as online information retrieval, emoji-enriched text generation and social media sentiment analysis [3, 4]. Typically, studies on emojis mainly focus on analyzing the semantics, usage or sentiment of emojis [1, 13, 2, 5, 6, 11, 12]. For example, Barbieri et al. [5] studied the semantic differences of emojis in different languages. Wijeratneet al. [14] proposed to incorporate the text descriptions of emojis to enhance the quality of emoji embedding. However, these approaches cannot mine the inherent relatedness between texts and emojis. In order to fill this gap, Barbieri et al. [3, 4] proposed a novel task to predict an emoji which is evoked by a plain tweet. They applied a hierarchical LSTM network to this task and achieved satisfactory performance. However, their methods simply aggregate all words together and can not distinguish the informative contexts from the uninformative ones in tweets. Thus, the performance may be sub-optimal. In addition, simply predicting a unique emoji for each message has two drawbacks. First, users often use different emojis within a message to express different opinions. For example, a tweet "tired after exam 😭, but the summer vacation comes! 🎉" uses multiple emojis to convey different emotions. Second, hot emojis often change over time [3]. For example, the emoji 🎄 may be frequently used during Christmas, but hardly used in other time. Therefore, using a fixed multi-class classification model is sub-optimal.

In order to solve these problems, we propose to predict multiple emojis evoked by tweets. We propose a hierarchical neural model with attention mechanism to address this task. Our model contains three modules. First, a character encoder is used to learn hidden representations of words from original characters using a CNN layer. In order to enrich the semantic information of word representations, we also concatenate the hidden vectors with pre-trained word embeddings. Since the user-generated tweet messages can be very noisy, learning word representations at character-level can capture useful patterns within words. Second, a word encoder is used to learn sentence representations by capturing both local and long-distance contextual information using a combination of Bi-LSTM and CNN. Since there are many uninformative words within tweets, attention mechanism is applied in this module to highlight the informative words. Third, an emoji classification module is used to predict the emojis. It will jointly classify all emojis to predict whether they are evoked by a tweet. Experimental results on real-world datasets show that our approach outperforms many baseline methods as well as humans in this task.

Our Approach

In this section, we introduce the details of our neural model. The architecture of our model is shown in Figure 1. We will introduce each module in detail.

Character Encoder

Usually, there are full of informal language in tweets, such as creative spellings, emoticons and slang. These words are often out-of-vocabulary, which may lead to weak representations of them. Since there are often specific patterns within these words, learning from original characters can enhance the representations of these words. In addition, character-level patterns can provide emotion clues,

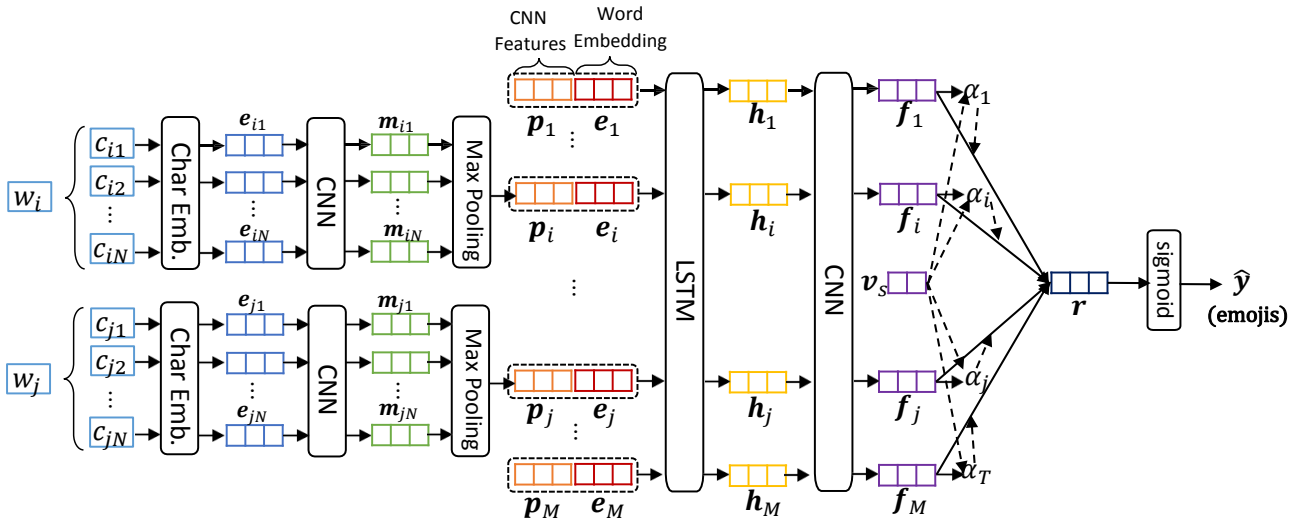

Figure 1: Architecture of our hierarchical model.

such as all-caps. Motivated by these observations, we introduce a character encoder to learn representations for each word in tweets. It contains two components. First, a character embedding layer is used to convert the character sequence of a word (denoted as $w_i = [c_{i1}, c_{i2}, ..., c_{iN}]$ for the i_{th} word, N represents the word length) into a sequence of vectors. The output of this layer is a vector sequence $\mathbf{E}_c = [\mathbf{e}_{i1}, \mathbf{e}_{i2}, ..., \mathbf{e}_{iN}]$.

Second, a character-level CNN layer with max pooling operation is explored to capture local information of characters and build hidden representations of each word, which can be computed by:

$$\mathbf{m}_{it} = \mathrm{ReLU}(\mathbf{W} \times \mathbf{e}_{i,(t-k):(t+k)} + \mathbf{b}), \quad (1)$$

$$\mathbf{p}_i = Maxpooling([\mathbf{m}_{i1}, \mathbf{m}_{i2}, ..., \mathbf{m}_{iN}]), \quad (2)$$

where $\mathbf{W}$ and $\mathbf{b}$ are the kernel weights and biases, $2k+1$ is the window size, $\mathbf{e}_{i,(t-k):(t+k)}$ denotes the concatenation of the character embeddings between step $t-k$ and $t+k$, ReLU is the activation function [7]. The output of this module is the hidden representations $\mathbf{P} = [\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_M]$ (M denotes the sentence length) for each word.

Word Encoder

The word encoder is used to learn the sentence representations at word-level. It contains four components. The first one is a word embedding layer. This layer is used to convert the word sequences (denoted as $W = [w_1, w_2, ..., w_M]$) into a sequence of vectors (denoted as $\mathbf{E}_w = [\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_M]$). It will be concatenated with the output $\mathbf{P}$ from the character encoder as the final word representations, which are denoted as $\mathbf{R} = [\mathbf{E}_w; \mathbf{P}]$.

The next one is a Bi-directional LSTM (Bi-LSTM) layer [8]. LSTM has been proven effective to capture long-distance information [9]. Since both past and future contexts are informative to build the representations of words, we employ Bi-LSTM to scan the input sequence in both directions. It takes the sequence $\mathbf{R}$ as input, and outputs the hidden representations $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_M]$ for words.

The third one is a word-level CNN. Local information is important to help to infer the meanings of sentences to predict emojis. For example, the phrase "birthday party" is often associated with the emoji 🎉. Therefore, we employ CNN to capture the local contexts in tweets. It takes the hidden representations of $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_M]$ as input, and outputs feature maps $F = [\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_M]$, which are formulated as follows:

$$\mathbf{f}_i = \mathrm{ReLU}(\mathbf{W}' \times \mathbf{h}_{i-k:i+k} + \mathbf{b}'), \quad (3)$$

where $\mathbf{W}'$ and $\mathbf{b}'$ are the CNN parameters, $\mathbf{s}_{i-k:i+k}$ represents the concatenation of the input sentence vectors between $i-k$ and $i+k$, ReLU is the activation function [7].

The fourth one is an attention layer. Usually contexts have different contributions to the overall sentence meanings. For example, the word "love" in the tweet "love my co-workers" is more informative than "my" to emoji prediction. Therefore, attention mechanism is incorporated to select important contexts. The attention weight α_i of the i_{th} word is formulated as:

$$\hat{\alpha}_i = \tanh(\mathbf{v}_s^T \mathbf{f}_i + \mathbf{b}_s), \quad (4)$$

$$\alpha_i = \frac{\exp(\hat{\alpha}_i)}{\sum_{t=1}^{T} \exp(\hat{\alpha}_t)}, \quad (5)$$

where $\mathbf{v}_s$, and $\mathbf{b}_s$ are parameters.

The representations $\mathbf{r}$ of the sentence is computed by the weighted sum of the hidden representations, which is formulated as follows:

$$\mathbf{r} = \sum_{i=1}^{M} \alpha_i \mathbf{f}_i. \quad (6)$$

Multilabel Emoji Classification

The classification module jointly predicts for each emoji whether it is evoked by the input tweet. Thus, the output probability for the i_{th} emoji is $\hat{y}_i = \sigma(\mathbf{W}_y^T \mathbf{s} + \mathbf{b}_y)$, where $\mathbf{W}_y$ and $\mathbf{b}_y$ are the parameters. The loss function $\mathcal{L}$ is the average binary crossentropy over all emojis, which is formulated as:

$$\mathcal{L} = -\frac{1}{K} \sum_{i=1}^{K} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)], \quad (7)$$

where y_i is the gold label of the i_{th} emoji, K is the number of emojis.

Experiments

Dataset and Experimental Settings

Our dataset for this task is retrieved via Twitter API during March of 2018. The raw collection of tweets contains 3,720,122 tweet messages. We select the top 30 frequent emojis in tweets as labels. The number of tweet containing emojis is 896,441. We randomly selected 500,000 tweets for model training, 50,000 for validation and 50,000 for test. In the final dataset, the percentage of tweets containing different number of distinct emojis is illustrated in Figure 2. In our dataset, 36.8% tweets contains more than one distinct emoji. The numbers of different emojis in the dataset in shown in Figure 3.

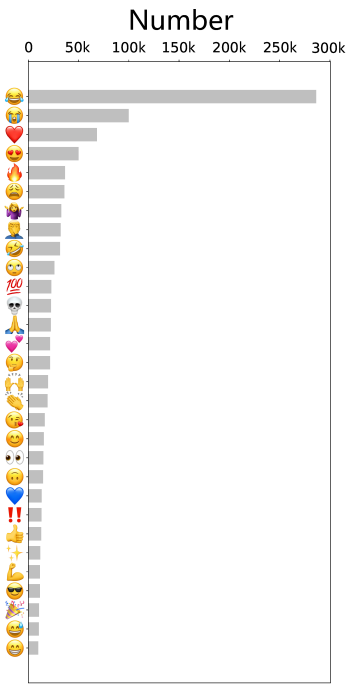

Figure 3: Statistics of the top frequent emojis in our dataset.

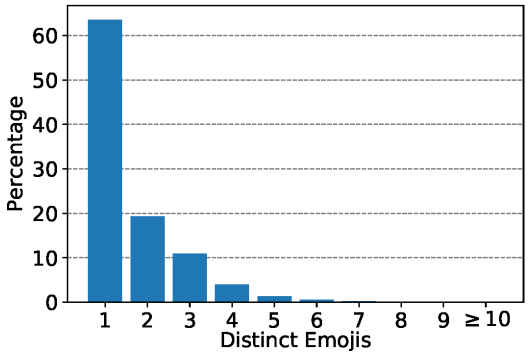

Figure 2: Percentage of tweets containing different number of distinct emojis.

The hyper-parameter settings in our experiments are summarized in Table 1. They are selected via cross validation on the training set. In order to mitigate overfitting, we apply dropout strategy at each layer in our network. We use the macro F-score over all emojis as the task metric. We repeat each experiment for 10 times and report the average results.

Hyper-parameter	**Value**
character embedding size	100
word embedding size	300
LSTM hidden state dimensions	2×200
CNN window size	3
CNN filters	300
dropout rate	0.2
batch size	256

Table 1: Hyper-parameters in our experiments.

Performance Evaluation

In this section, we compare the performance of our approach with several baselines. The methods to be compared include:

- SVM, using K independent bag-of-word SVM models to predict for each emoji.
- CNN, using CNN as the word encoder only and predict all emojis jointly.
- LSTM, using LSTM as the word encoder.
- LL, a hierarchical model with LSTM in both character and word encoders [10].
- CLC, a variant of our proposed model without attention mechanism.
- CLCA, our proposed model.

In addition, we also compare the performance of humans by inviting a group of volunteers to solve the same problem. We conduct experiments using different amount of training data. The experimental results are illustrated in Figure 4. According to the results, several important observations are as follows:

(1) neural methods consistently outperform SVM. Since the SVM models only take the bag-of-word features as input, they cannot capture the contextual information in tweets, which are usually important to predict emojis.

(2) hierarchical models (CLCA and LL) outperform the flatten models (CNN and LSTM). Character information is often useful to capture specific information. For example, the

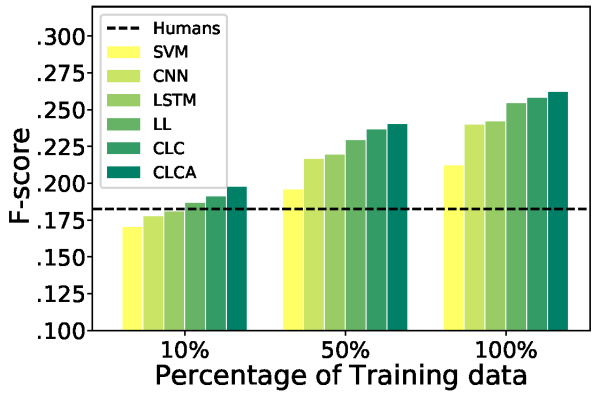

Figure 4: Performance of different automatic methods and human annotators.

word "NOOOOT" which contains repeated all-caps is an indication of strong sentiment. Therefore, incorporating character information is beneficial for inferring the emojis.

(3) our proposed model (CLC and CLCA) consistently outperforms other baeslines such as LL. It shows that using the combination of CNN and LSTM can achieve a better performance than using LSTM only. It may be because that combining CNN and LSTM can capture both local and long-range contexts, which is useful to mine the relationships between texts and emojis. Besides, applying attention mechanism can further improve the performance of our model. This is probably because that focusing on important words in tweets is useful to predict the emojis evoked by tweets.

(4) automatic methods can outperform humans if training data is sufficient. It may be because that an automatic neural model can predict emojis more objectively than human annotators. In our approach, it is cheap to obtain a large amount of self-labeling data. Therefore, a robust model can be easily constructed.

Qualitative Analysis

The F-score performance of our best-performance model CLCA and the baseline models CLC and LL is shown in Figure 5. Several qualitative findings are as follows: First, our model can outperform baselines in most emoji categories and can achieve a good performance when predicting many frequent emojis such as 😂 and ❤️. It's probably because that the training data of these emojis is sufficient. Second, the performance of some infrequent emojis such as ‼️ and 🎉 is also satisfactory. It may be because these emojis are often used to express specific meanings. For example, the emoji 🎉 is often an indication of excitement. Therefore, these emojis can be predicted more easily. Third, we find it's interesting that sad emojis such as 😭 and 😩 are harder to predict than happy ones. It may be because sad emotions can often be implicit, which are hard to be inferred by plain messages.

Conclusion

In this paper, we introduce a neural approach for multi-label emoji prediction in tweets. We propose a hierarchical neural model with attention mechanism to address this task. Our model contains three modules, a character encoder to learn hidden representations of words using CNN, a word encoder to learn sentence representations using a combination of CNN and LSTM, an emoji classifier to predict emojis for tweets. Attention mechanism is incorporated into our model, which aims to build high quality sentence representations by highlighting important contexts. Besides, since the dataset in our experiments is automatically constructed, no manual annotation is needed in our approach. The experimental results validate the effectiveness of our approach.

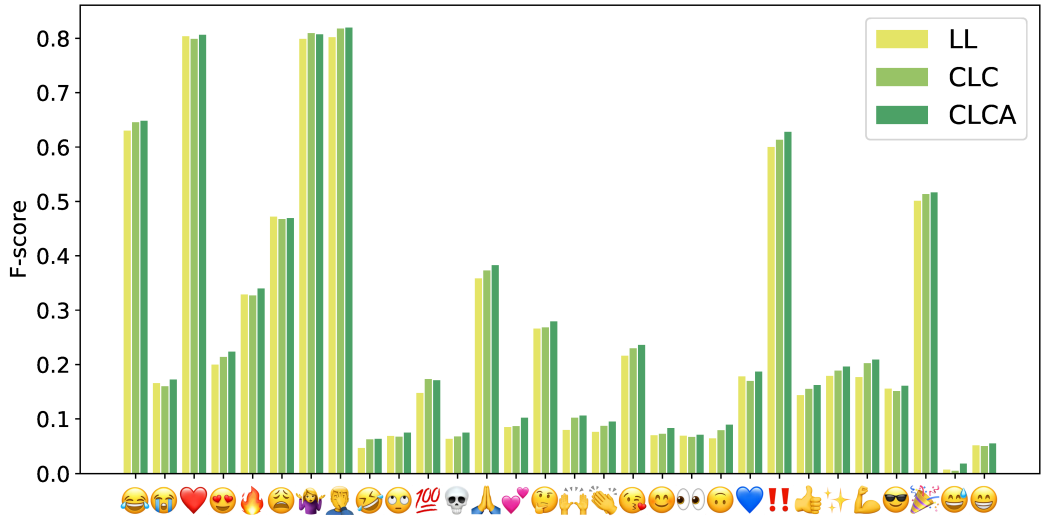

Figure 5: F-score performance of different models over all emojis.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB0800402 and in part by the National Natural Science Foundation of China under Grant U1705261, Grant U1536207, Grant U1536201 and U1636113.

REFERENCES

1. Sho Aoki and Osamu Uchida. 2011. A method for automatically generating the emotional vectors of emoticons using weblog articles. In *Proc. 10th WSEAS Int. Conf. on Applied Computer and Applied Computational Science, Stevens Point, Wisconsin, USA*. 132–136.
2. Francesco Barbieri, Luis Espinosa Anke, and Horacio Saggion. 2016. Revealing Patterns of Twitter Emoji Usage in Barcelona and Madrid.. In *CCIA*. 239–244.
3. 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*. Association for Computational Linguistics, 105–111. `http://aclweb.org/anthology/E17-2017`
4. 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)*. Association for Computational Linguistics, New Orleans, LA, United States.
5. Francesco Barbieri, German Kruszewski, Francesco Ronzano, and Horacio Saggion. 2016a. How

cosmopolitan are emojis?: Exploring emojis usage and meaning over different languages with distributional semantics. In *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 531–535.

6. Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. 2016b. What does this Emoji Mean? A Vector Space Skip-Gram Model for Twitter Emojis.. In *LREC*.
7. Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*. 315–323.
8. Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5-6 (2005), 602–610.
9. Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.
10. Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Finding function in form: Compositional character models for open vocabulary word representation. *arXiv preprint arXiv:1508.02096* (2015).
11. Nikola Ljubešić and Darja Fišer. 2016. A global analysis of emoji usage. In *Proceedings of the 10th Web as Corpus Workshop*. 82–89.
12. Hannah Miller, Daniel Kluver, Jacob Thebault-Spieker, Loren Terveen, and Brent Hecht. 2017. Understanding emoji ambiguity in context: The role of text in emoji-related miscommunication. In *11th International Conference on Web and Social Media, ICWSM 2017*. AAAI Press.
13. Petra Kralj Novak, Jasmina Smailovic, Borut Sluban, and Igor Mozetic. 2015. Sentiment of Emojis. *PLOS ONE* 10, 12 (2015).
14. Sanjaya Wijeratne, Lakshika Balasuriya, Amit Sheth, and Derek Doran. 2017. A Semantics-Based Measure of Emoji Similarity. In *2017 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. ACM, ACM, Leipzig, Germany. DOI: http://dx.doi.org/10.1145/3106426.3106490