LIS at SemEval-2018 Task 2: Mixing Word Embeddings and Bag of Features for Multilingual Emoji Prediction

Gaël Guibon
LIS UMR 7020
Aix Marseille Université
CNRS
Caléa Solutions
gael.guibon@lis-lab.fr

Magalie Ochs
LIS UMR 7020
Aix Marseille Université
CNRS
magalie.ochs@lis-lab.fr

Patrice Bellot
LIS UMR 7020
Aix Marseille Université
CNRS
patrice.bellot@lis-lab.fr

Abstract

In this paper we present the system submitted to the SemEval2018 task2 : Multilingual Emoji Prediction. Our system approaches both languages as being equal by first; considering word embeddings associated to automatically computed features of different types, then by applying bagging algorithm RandomForest to predict the emoji of a tweet.

1 Introduction

Emojis were first used to emphasize conversations before becoming representations of specific emotions, objects or ideas. They are now used in almost every social medium and conversation devices, such as messenging applications or even emails[1].

Tweets and their emoticons were used as labels to predict polarity at first (Pak and Paroubek, 2010). However, emojis are not used the same way as emoticons in messaging applications. They can convey further information, even more when combined. The advantage of emojis is that they are becoming more standardized, even though existing emojis are still growing quickly[2]. This is why emoji prediction is a relatively new task. It can be considered as a composite task mixing emotion prediction for face emojis, aspect/subject detection for object emojis, and other metadata prediction for more abstract emojis, representing ideas for instance.

This year, SemEval started the first emoji prediction task (Barbieri et al., 2018). It consists of a multiclass classification task for a total of 20 possible classes, *i.e.* emojis. This task is interesting in several ways. Firstly, it is a relatively new task that only a few studies did focus on. Secondly, it is quite important not only for research, but also for companies willing to embrace the current trend of social network and interaction analysis. Both are important topics for Natural Language Processing (NLP) and Information Retrieval (IR).

Our system obtained good results (63.65% f1-score) while using the trial dataset, and lower results (13.53% f1-score) on the test dataset. Because this pattern occurred for both English and Spanish, and for all participants, we try to explain it.

The paper is organized as follows: we first summarize the existing work related to this task and to our approach (Section 2). Then we present what we identified as the most challenging areas from this task and the dataset used (Section 3). We go on by describing our system (Section 4) and detailing the pre processing and prediction steps. Finally, we conclude by discussing the performance limits and show the benefits of our participation in this task (Section 5).

2 Related Work

Several research studies focus on emoji prediction. Most of them use word embeddings in order to do a multiclass emoji prediction. At the beginning, images were used instead of text as the source of emoji prediction (Cappallo et al., 2015). Eisner (Eisner et al., 2016) used embeddings based on emoji description in the Unicode[3] list, such as *smiling face with heart eyes*. They obtained 85% accuracy in their classification of emoji descriptions, predicting several keywords for one emoji. Xie (Xie et al., 2016) trained neural networks on Weibo[4] to predict 10 possible emojis in conversations with 65% accuracy for the 3 mostly used emojis. Barbieri (Barbieri et al., 2017) then

[1] http://cdn.emogi.com/docs/reports/2015_emoji_report.pdf
[2] https://goo.gl/jbeRYW
[3] http://unicode.org/ emoji/charts/full emojilist.html
[4] http://www.weibo.com/

predicted 20 emojis in millions of tweets using LSTM (Hochreiter and Schmidhuber, 1997) and obtained 65% f1-score for the 5 most used emojis. Felbo (Felbo et al., 2017) tackled emoji prediction by LSTM with 43.8% accuracy for the top 5 emojis, while using emoji vectors to help detect sarcasm. In our recent work we considered another approach with 84.48% weighted F1-score using multi-label emoji prediction of 169 sentiment related emojis in real private messages (Guibon et al., 2018).

3 Task Specific Difficulties

Be it in English or Spanish, the proposed task has specific difficulties. Each of these difficulties represents challenges and obstacles for the classifier to make a good prediction.

First, the dataset is made of 20 classes of different types and concepts. Some are related to pure emotions ❤️, facial expressions of emotions 😂, or even classes representing objects 📷 or ideas ✨. Those different classes may sometimes appear in a same context (❤️ 💜, 💕, and 💙 for instance), even though the dataset was selected to only keep tweets with only one emoji.

Second, tweets are not private short messages. This means that some tweets are even difficult to understand for humans. This is the case for reaction tweets to a certain hashtag or social event. The appreciation of the event is totally dependent on the user's subjective point of view. Thus, it is also the case for the resulting emoji associated to the message. Other types of tweet-emoji associations, such as advertisements, are not even humanly predictable.

Third, the dataset is really unbalanced, which has become quite common in real applied classification. However, it still represents a challenge when associated to the two previous difficulties. Taken together, they make emoji prediction quite difficult, especially for tweets, which justifies even more the necessity for this task.

Two datasets[5] were used for emoji prediction in tweets: 500 000 tweets in training and 50 000 as trial and test for English, 100 000 tweets in training and 10 000 as trial and test for Spanish. Each dataset was made of tweets containing only one emoji between a set of 20 most frequent emojis from tweets containing only one emoji.

The emoji set only contains positive or neutral emojis, making a sentiment analysis approach less relevant, but we still kept using polarity scores in order to include the intensity of the polarity as a feature.

4 System

4.1 Preprocessing

Cleaning. To prepare the data we first cleaned tweets by removing trailing three dots, user mentions and urls. Then we used Spacy[6] to apply lemmatization and part-of-speech tagging (PoS).

Word Representation. For data representation, we compared different approaches for text vectorization. We first did a text representation using *FastText* (Bojanowski et al., 2016) but did not obtain an overall gain in the prediction in comparison to *Word2Vec* (Mikolov et al., 2013). We used *Word2Vec* in its Gensim[7] (Rehurek and Sojka, 2010) implementation with the following hyper parameters:

- Architecture: Continuous Bag-of-Words
- Batch size: 32
- Minimum count: 1
- Embedding size: 50 or 300
- Iterations: 100

The minimum count was set to 1 in order to better capture rare items from really small tweets, and the Continuous Bag of Words (CBOW) architecture was prefered after empirical tests to determine if it was useful to use it or not. The best text vectorization was obtained using live-trained embeddings, without using external pre-trained embeddings, even though we trained word embeddings and character embeddings on millions of tweets to obtain better representation, and also used existing pre-trained embeddings (Barbieri et al., 2016). This is certainly due to the overlap between the training and the trial set. Thus the local vectorization is more representative to find already known contexts. Varying the size of the embedding matrix E did not show major improvements for the following prediction, whether its dimension was $d300$ or $d50$. Thus, we chose a dimension of $d50$ to train faster. Tweets are represented as the *mean* of each word embedding vectors, allowing

[5] https://github.com/fvancesco/Semeval2018-Task2-Emoji-Detection

[6] https://spacy.io/

[7] https://radimrehurek.com/gensim/about.html

the same size ($d50$) for each tweet final embedding vector.

Computed Features. In addition to the embedding vectors, we computed several features represented as a feature vector F: binary features for the presence of a question or an interrogation mark, and their repetitions, another boolean feature for the usage of Title Case. Numerical counts were also added: word count, character count, average token length, number of nouns, adjectives, adverbs, interjections and verbs. Polarity prediction was also added by using SentiStrength (Thelwall et al., 2010) positive and negative scores. The advantage being that we then have polarity intensity, so it could be useful even if all 20 emojis are neutral or positive.

Finally, this feature vector F of dimension $d23$ was added to each embedding matrix E along the columns axis. The matrix is as follow: each row represents one tweet, and each column a feature. Each tweet information being represented by $E + F$. This means that before concatenation, a row (*i.e.* a tweet) has 50 columns, and after concatenation, it has 73 columns.

This pre-processing approach was used for all data separately, meaning that we based all our tests while training on the training set, and testing on the trial set. We used this approach for both English and Spanish.

4.2 Prediction

The system used was chosen after trying multiple approaches using the training set for train the model and the trial set to obtain macro F1-score. We explored multi-class RBF-SVM with gaussian distance function, LSTM network (3 LSTM layers with 64 unit cells, 0.5 dropout, then softmax layer) and decision tree based algorithms (XGBoost, decision tree, RandomForest). Decision tree based algorithms always gave us better results to take into account all classes during prediction. The number of systems were limited to 2, so we applied sightly different approaches.

In our system we used RandomForest with 700 estimators chosen empirically in order to predict emojis. To automatically find the best parameters we used a grid search with cross validation strategy for specific parameters visible in Table 1. The best parameters found were quite similar to the default one from the Scikit-Learn API except for the balanced subsample class weight. We also tried setting the class weight manually to deal with unbalanced dataset. We gave more weight (5) to the 3 majority classes ❤️😍😂 and left the other classes to 1, without improving the results. The maximum depth for each tree was then set to None because we believe a bagging approach such as RandomForest with a number of estimators higher than the targetted classes can compensate overfitting issues coming from a higher complexity of each estimator.

Max Depths	20, 100, 200
Min Samples Splits	2, 5
Min Samples Leafs	1, 4
Max Features	sqrt, log2, None
Criterions	'gini', 'entropy'
Class Weights	None, 'balanced' 'balanced subsample'

Table 1: Grid search for RandomForest parameters.

The two submissions vary slightly, but are still the same system.

Version 1. On the one hand, data were scaled from 0 to 1 and we used a $\log_2$ parameters and χ^2 feature selection to minimize the number of features. This is based on the assumption that useful data in the word embeddings should be scaled before being concatenated with the features vector, then only embeddings and useful computed features should be used.

Version 2. On the other hand, we did not scaled any data nor limited the number of features, as suggested by the grid search.

According to feature importance scores from the classifier (Table 2), the best computed features were the average token length, the character and word counts, and the number of uppercases. The other features have minor impact even though PoS tag counts follow the top five features.

1	averageTokenLength (0.016)
2	charCount (0.015)
3	wordCount 0.012)
4	upperCharCount (0.011)
5	nounCount (0.009)
...	

Table 2: Top five computed features.

We first used only embeddings to predict, then predicted using concatenated embeddings and computed features vectors. The latter improved the overall prediction, which can also be seen by the feature importance scores.

We managed to obtain 63.65% macro f1-score on English, and 84.13% macro f1-score on Span-

ish while predicting on the official trial corpus. The English classification report is visible in Table 3. Also, the model obtained 61.92% accuracy on english and could be upgraded by sometimes choosing one of the best probabilities from each prediction according to the Mean Reciprocal Rank (MRR) score of 0.7126.

Emo	P	R	F1
❤️	0.42	0.92	0.58
😍	0.82	0.51	0.63
😂	0.58	0.76	0.66
💕	0.97	0.44	0.60
🔥	0.73	0.62	0.67
😊	0.97	0.44	0.61
😎	0.95	0.45	0.61
✨	0.94	0.46	0.62
💙	0.96	0.44	0.61
😘	0.97	0.43	0.60
📷	0.70	0.68	0.69
🇺🇸	0.86	0.61	0.71
☀️	0.76	0.51	0.61
💜	0.98	0.42	0.59
😉	0.99	0.48	0.64
💯	0.96	0.48	0.64
😁	0.97	0.46	0.63
🎄	0.87	0.68	0.76
📸	0.89	0.51	0.65
😜	0.99	0.45	0.62
Avg.	0.76	0.62	0.62

Table 3: Precision, Recall, F-measure for each emoji on the trial set.

However, our system obtained poor results once applied on the official test set, with only 13.528% macro f1-score on English, and 8.808% macro f1-score on Spanish.

Performance decrease in test set. An overall drastic performance decrease was shown while applying the model on the test set. We believe this is due to multiple factors. First, as we have no means to identify very difficult tweets for which even humans could not predict emoji (see Section 3), it is difficult to know to what extent the model generalized well. Of course, by comparing our approach results with other ones, we know that the model or the approach should be improved in order to better take into account all classes, as it is visible in the test set confusion matrix (Figure 1).

Another element explaining the major performance decrease is the presence of overlapping elements between the trial set and the training set

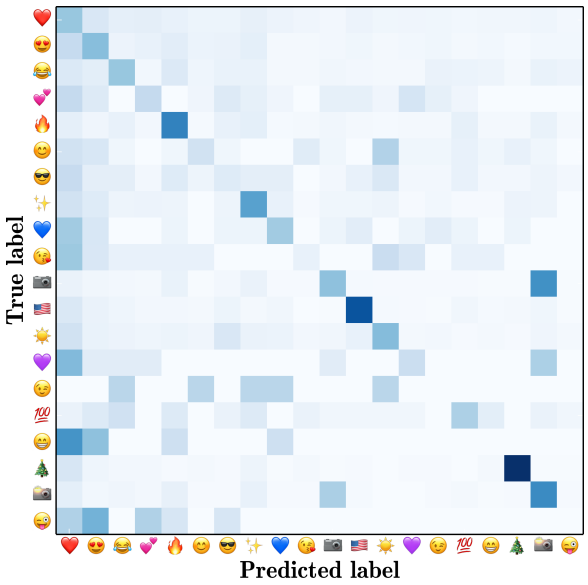

Figure 1: Confusion matrix from official test results.

that misleaded parameters tuning. Even though, we think a text representation enhancement is necessary, as this approach finally gave poor results.

5 Conclusion

In this paper we described the system we submitted to the SemEval-2018 Task 2 for Multilingual Emoji Prediction. The system presented uses text vectorization through word embeddings associated to a computed-features vector in order to represent each tweet by their polarity intensity and metrics. The classification is then done by using decision tree based algorithm for understanding, with bagging technique for better generalization to match the goal of macro F1-score metric. With this system we wanted to have a generic system for both languages without specific parameters for each language.

The system obtained good results on the trial set but the performances decreased drastically when applied to the test set. Even though this pattern was shown through all participants' systems, ours finally obtained poor results on the test set. We believe it is necessary to further process the data in order to identify recurrent difficult cases, such as really short and commons tweets. A more robust representation of each tweet is also required.

Finally, the python code used for this task is available on github[8].

[8] https://github.com/gguibon/SemEval2018-Task2-MultilingualEmojiPrediction

References

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.

Francesco Barbieri, German Kruszewski, Francesco Ronzano, and Horacio Saggion. 2016. How cosmopolitan are emojis?: Exploring emojis usage and meaning over different languages with distributional semantics. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 531–535. ACM.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*.

Spencer Cappallo, Thomas Mensink, and Cees G.M. Snoek. 2015. Image2emoji: Zero-shot emoji prediction for visual media. pages 1311–1314. ACM Press.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description. *arXiv preprint arXiv:1609.08359*.

Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. 2017. Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. *arXiv preprint arXiv:1708.00524*.

Gaël Guibon, Magalie Ochs, and Patrice Bellot. 2018. Emoji recommendation in private instant messages. In *Proceedings of the 2018 ACM symposium on Applied computing*, pages 1810–1813. ACM.

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

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

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10.

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. 2010. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology*, 61(12):2544–2558.

Ruobing Xie, Zhiyuan Liu, Rui Yan, and Maosong Sun. 2016. Neural emoji recommendation in dialogue systems. *arXiv preprint arXiv:1612.04609*.