Tübingen-Oslo at SemEval-2018 Task 2: SVMs perform better than RNNs at Emoji Prediction

Çağrı Çöltekin
Department of Linguistics
University of Tübingen, Germany
ccoltekin@sfs.uni-tuebingen.de

Taraka Rama
Department of Informatics
University of Oslo, Norway
tarakark@ifi.uio.no

Abstract

This paper describes our participation in the SemEval-2018 task Multilingual Emoji Prediction. We participated in both English and Spanish subtasks, experimenting with support vector machines (SVMs) and recurrent neural networks. Our SVM classifier obtained the top rank in both subtasks with macro-averaged F1-measures of 35.99 % for English and 22.36 % for Spanish data sets. Similar to a few earlier attempts, the results with neural networks were not on par with linear SVMs.

1 Introduction

Emojis are graphical symbols that represent an idea or emotion. The use of emojis has become popular over the last decade, particularly in informal communication in the social media. Their popularity kindled a recent interest in investigating many aspects of emojis, including their interaction with natural language (e.g., Barbieri et al., 2016, 2017; Felbo et al., 2017; Kralj Novak et al., 2015). Although the emojis are presumably language-independent, their use typically goes together with linguistic text. In this context, the SemEval 2018 task 2, Multilingual Emoji Prediction (Barbieri et al., 2018), aims predicting the emoji from the surrounding micro-blogging (Twitter) text for English and Spanish.

The task at hand is to predict a label, an emoji, from a short text that it accompanies. This is essentially a text/document classification problem, and shares many aspects of other text classification problems such as topic classification, sentiment analysis, language identification and authorship attribution – just to name a few. Although each of these problems have some task-specific aspects, the same models can be used for all of them. In this study, we experiment with and compare two well-known methods: support vector machines (SVMs) with bag of word/character n-gram features and recurrent neural networks (RNNs) with word and character sequences as input. The methods and implementations are similar to our earlier attempts in other text classification tasks (Çöltekin and Rama, 2016; Rama and Çöltekin, 2017; Çöltekin and Rama, 2017).[1] In the remainder of this paper, we describe our methods and experiments, present and discuss our results.

2 Experiments and Results

We participated in both subtasks using the same architectures. However, we trained and tuned the model parameters on each data set separately. The training set for the competition consisted of 500 000 tweets for English and 100 000 tweets for Spanish subtask. The data sets contained most frequent 20 emojis for English and 19 emojis for Spanish. Joining late to the party, our training set consisted of 485 151 English tweets, and 97 765 Spanish tweets, since about 3 % of the tweets were not available by the time we crawled them. As presented in Figure 1, the label distribution is similar and quite skewed for both languages. We included pre-processing steps of case normalization and discarding low-frequency features as part of our hyperparameter optimization. In all our experiments, we use only the data supplied by the organizers. We did not use any external sources (e.g., pre-trained word embeddings), nor did we perform any further linguistic processing (e.g., POS tagging, or parsing). The test size for English and Spanish is 50 000 and 10 000 respectively.

2.1 Support Vector Machines

The best results obtained in the shared task are based on multi-class (one-vs-rest) linear support

[1] The source code of our implementation is available at https://github.com/coltekin/emoji2018.

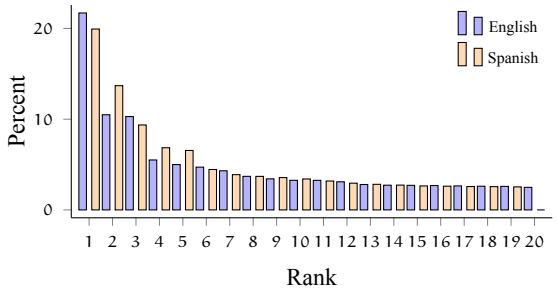

Figure 1: Label distribution in both data sets. Ratio of each label is plotted against its rank. Note that the emojis sharing the same rank are not necessarily identical in both languages.

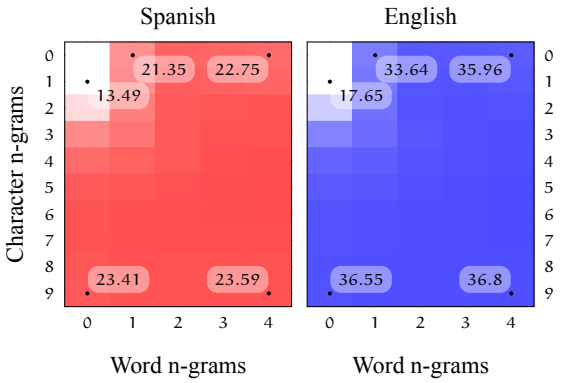

Figure 2: The effect of maximum character and word n-gram size combinations to F1-measure. Darker shades indicate higher F1-measure.

vector machines (SVM). We use 'bag of n-grams' as features, combining both character n-grams and word n-grams of different sizes, weighted by sublinear TF-IDF scaling applied globally to all n-grams (character and word n-grams with varying sizes). Although we also experimented with logistic regression and random forests using the same feature set, the results were consistently inferior to the SVMs. Therefore, we will not discuss the results of logistic regression and random forests. The models discussed in this section were implemented with scikit-learn package (Pedregosa et al., 2011) using liblinear back end (Fan et al., 2008).

We optimized the models for best macro F1-score on each language data set through a grid search using 5-fold cross validation. The hyperparameters considered during optimization were maximum character/word n-gram size, case normalization, minimum document frequency threshold for excluding low-frequency features, and SVM margin (or regularization) parameter 'C'. Although there has been other parameter settings with competitive scores, we used maximum character n-grams size of 6, maximum word n-gram size of 4, minimum document frequency threshold of 2, SVM parameter C of 0.10, and we case normalized only word (not character) n-grams. Our submitted system achieved 36.55 precision, 36.22 recall and 35.99 F1-score on the English test set, and 23.49 precision, 22.80 recall and 22.36 F1-score on the Spanish test set. These figures were about 1 % lower than the figures we obtained in 5-fold cross validation results on the training data.

Figure 2 presents the effects of character and word n-grams of different sizes. For all results presented in Figure 2, n-grams from size 1 up to the indicated number are included as features. Although both combining character and word n-grams, and larger n-gram sizes increase the performance, the gains from higher n-gram values are rather small. The effects of other hyperparameters are smaller. In general, however, excluding features based on frequency seems to hurt the performance. Case normalization is useful if applied to word n-grams, but its effects are often negative if it is applied to both character and word n-grams. The optimum regularization parameter 'C' is stable over both languages and different training sizes.

2.2 Recurrent Neural Networks

Gaining popularity relatively recently, neural models are another common approach to text classification. Fully-connected networks are computationally impractical. However, convolutional networks (CNNs) and recurrent neural networks (RNNs) offer reasonably efficient computation, as well as better modeling of sequences. RNNs, particularly gated RNNs, have been used in many diverse natural language processing tasks successfully, and text classification is not an exception.

Our neural model includes two bidirectional RNN components: one taking a sequence of words as input and another taking a sequence of characters as input. The recurrent components of the network builds two representations for the text (one based on characters, the other based on words), the representations are concatenated and passed to a fully connected softmax layer that assigns an emoji to the document based on the RNN representations. Since the tweets are relatively short, we did not truncate the input documents. For both character and word inputs, we used embedding layers before the RNN layers. All neural network experiments were implemented with Tensorflow (Abadi et al., 2015) using Keras API (Chollet et al., 2015).

Although the history/context is not a parameter for recurrent networks, the architecture has many hyperparameters. We optimized the hyperparameters of the architecture through a random search for the embedding size of both characters and words, the hidden representation size of the RNN cells, the dropout parameter for each component of the network, frequency threshold for excluding features, RNN architecture, GRU (Cho et al., 2014) or LSTM (Hochreiter and Schmidhuber, 1997), and case normalization. For the RNN models, we used a random training–validation split (80 %–20 % for Spanish, and 90 %–10 % for English) during hyperparameter tuning. We used early stopping based on macro F1-measure, and picked the epoch with the best F1-measure for each hyperparameter setting. Besides these parameters – used for systematic random search – we also experimented with deeper architectures, both by stacking RNNs and by multiple fully-connected layers. Deeper networks, however, yielded worse results.

We obtained F1-scores of 33.02 % for English data and 17.98 % for the Spanish data on the (randomly split) development set. For both subtasks, we submitted results with the hyperparameter setting that worked best on the English data set (although it yielded a slightly lower F1-score than the best one obtained for Spanish). For both languages, the RNN results submitted used a model with embedding layers of size 32 (for characters) and 128 (for words). In the case of bidirectional GRU networks we used hidden units of sizes 32 and 128 for character and word input, respectively, minimum frequency threshold of 4 for characters and 1 for words, dropout parameter of 0.50 at the embedding layers and 0.10 at the RNN layers, and no case normalization.

2.3 Effect of training set size

The performance with different training set sizes is an important consideration in model choice. Furthermore, since the training set sizes for the two languages in present study are different, it is also be a plausible explanation for the fact that substantially lower performance of both models on the Spanish task. To shed light into these two issues, we present incremental results on (only) the English data set. In this experiment, we randomly set aside 10 % of the English training data for testing, we split the remaining 90 % into 10 splits, and train both systems by starting with one of splits, and incrementally adding another one in each iteration.

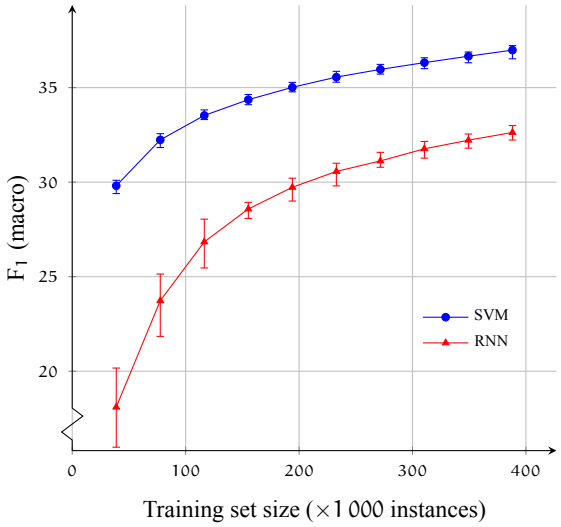

Figure 3: Learning curve for the SVM and RNN models on the English training set. The error bars indicate maximum and minimum values in 10 trials.

Figure 3 shows the F1-score against training set size, for both SVM and RNN models.

3 Discussion and conclusions

In this paper, we described our submitted systems at SemEval-2018 Task 2 on Multilingual Emoji Prediction. Besides providing details on our systems, this paper also intends to provide a comparison between two text classification methods: RNNs and linear SVMs. The comparison is motivated by the fact that, despite their popularity and argued superiority, we and others found linear models, particularly SVMs, yield better results than (deep) neural models in a series of other text classification tasks (e.g., Çöltekin and Rama, 2016; Rama and Çöltekin, 2017; Çöltekin and Rama, 2017; Medvedeva et al., 2017).

One plausible explanation is the fact that neural networks typically require more data to train. Indeed, the previous shared tasks cited above often provided modest-size training sets, mainly due to the cost of labeling. Emoji classification task has an advantage in this respect as the labeling is relatively cheap compared to many other text classification tasks. As a result, at least for English, the shared task included a rather large training set. However, our current findings also indicate that the linear SVMs still perform better than the RNN counterparts. Although the results presented in Figure 3 indicate that more data is, indeed, helpful for RNNs, the performance gap in favor of SVMs persists. Another interesting (but expected due to model complexity) observation in

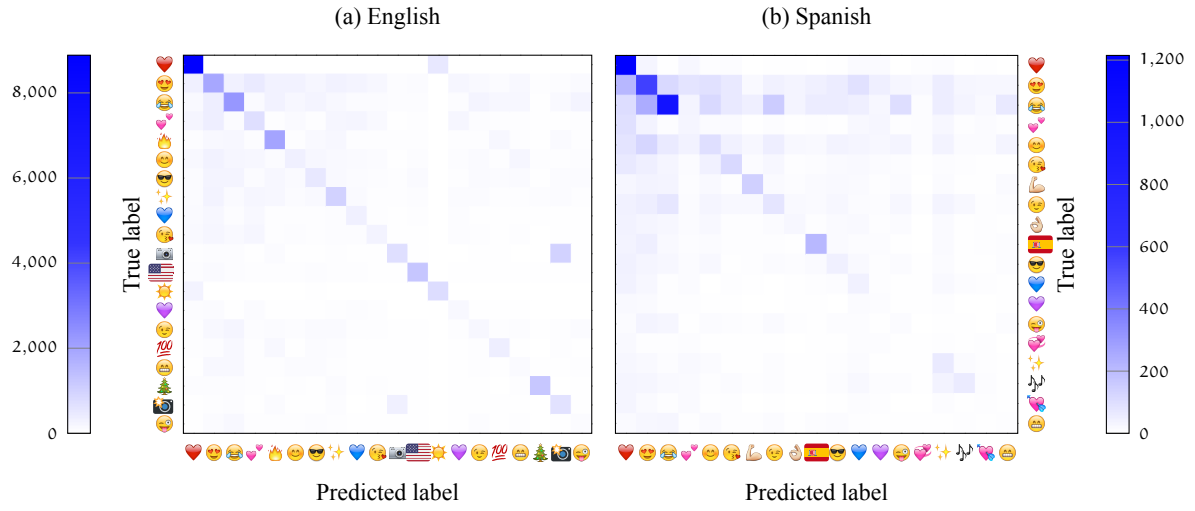

Figure 4: Confusion matrices for both data sets. The labels are sorted by frequency.

Figure 3 is that the RNNs also exhibit larger variation, especially with smaller data sizes.

Our findings seem to contradict with the majority of recent NLP literature, where RNNs are often claimed to be superior to linear models, and emoji classification is not an exception (e.g., Barbieri et al., 2017). Part of this impression comes from the fact that, in most studies, the linear baselines used in comparison are simple bag-of-words models. As words in a text are not independent, simple bag-of-words is deemed to fail. The simple addition of word n-gram features, however, circumvents this problem to a large extent, enabling the linear models to capture some local dependencies. RNNs, however, still have a potential advantage since they can, at least in theory, capture long-range dependencies as well. However, it seems either local dependencies are enough in many text classification tasks, or the data sets are (still) small for RNNs to generalize over useful long-range dependencies. Furthermore, character n-gram features are also useful, particularly for morphologically rich languages, as they also capture information present in sub-word units. Although including many overlapping character and word n-gram features result in large feature vectors, the sparse implementations of these models are computationally feasible and easy to tune – often more than corresponding deep neural network models.

A curious finding from our experiments is that despite the language-agnostic nature of our methods, both models yielded a rather large performance difference (13.63 % F1-measure on the test set) between English and Spanish. The possible explanation based on training set size is not supported by the experiments presented in Section 2.3. Figure 3 shows that, at about the training set size of Spanish data (100 000 instances), one can obtain about 32 % F1-score on the English data set, which is substantially higher than the best test and development set results we obtained using the full training data for Spanish (22.36 % and 23.59 % respectively). Hence, the difference is likely to be either due to differences between the languages, or due to some inherent confusability of the emojis in the Spanish data set. The confusion matrices in Figure 4 indicate higher majority class bias for Spanish. More experiments are needed for a better understanding of the differences.

3.1 Future directions

Past research has found that ensemble methods that combine multiple classifiers yield better performance compared to each individual classifier (Malmasi and Dras, 2015). Besides the differences in the learning algorithms, the models we compare in this work exploit rather different types of information. Hence, a combination of classifiers may result in better performance. Even though we did not experiment with ensemble methods in this work, the number of test instances that were predicted correctly by one of the models (but not by both) was 17.28 % and 19.95 % for English and the Spanish data respectively, indicating a promising upper bound for an ensemble approach.

Although we did not use any external resources in this task, another potential source of improvement is to use external information (e.g., embeddings or cluster labels) extracted from large unlabeled texts.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

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, Valencia, Spain. 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, Francesco Ronzano, and Horacio Saggion. 2016. What does this emoji mean? a vector space skip-gram model for twitter emojis. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France. European Language Resources Association (ELRA).

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

François Chollet et al. 2015. Keras. `https://github.com/keras-team/keras`.

Çağrı Çöltekin and Taraka Rama. 2016. Discriminating similar languages with linear SVMs and neural networks. In *Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial3)*, pages 15–24, Osaka, Japan.

Çağrı Çöltekin and Taraka Rama. 2017. Tübingen system in VarDial 2017 shared task: experiments with language identification and cross-lingual parsing. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 146–155, Valencia, Spain. Association for Computational Linguistics.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874.

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. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1615–1625, Copenhagen, Denmark. Association for Computational Linguistics.

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

Petra Kralj Novak, Jasmina Smailović, Borut Sluban, and Igor Mozetič. 2015. Sentiment of emojis. *PLOS ONE*, 10(12):1–22.

Shervin Malmasi and Mark Dras. 2015. Language identification using classifier ensembles. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pages 35–43.

Maria Medvedeva, Martin Kroon, and Barbara Plank. 2017. When sparse traditional models outperform dense neural networks: the curious case of discriminating between similar languages. In *Proceedings of the Fourth Workshop on NLP for Similar Languages, Varieties and Dialects (VarDial)*, pages 156–163, Valencia, Spain. Association for Computational Linguistics.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Taraka Rama and Çağrı Çöltekin. 2017. Fewer features perform well at native language identification task. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 255–260, Copenhagen, Denmark. Association for Computational Linguistics.