# UMDuluth-CS8761 at SemEval-2018 Task 2: Emojis: Too many Choices? 😉

**Dennis Asamoah Owusu & Jonathan Beaulieu**
Department of Computer Science
University of Minnesota Duluth
Duluth, MN 55812 USA
`{asamo012,beau0307}@d.umn.edu`
`https://github.com/derpferd/semeval-2018-task2/`

## Abstract

In this paper, we present our system for assigning an emoji to a tweet based on the text. Each tweet was originally posted with an emoji which the task providers removed. Our task was to decide out of 20 emojis, which originally came with the tweet. Two datasets were provided - one in English and the other in Spanish. We treated the task as a standard classification task with the emojis as our classes and the tweets as our documents. Our best performing system used a Bag of Words model with a Linear Support Vector Machine as its' classifier. We achieved a macro F1 score of 32.73% for the English data and 17.98% for the Spanish data.

## 1 Introduction

An AI system that can associate text with appropriate emojis could be useful for generating content that is sparkled with emojis among other uses (Barbieri et al., 2017). Given only the text from a tweet in English or Spanish, the SemEval (Barbieri et al., 2018) task was to determine the emoji that was in the original tweet. To learn how users associate emojis with text, a dataset comprised of 489,609 tweets in English and 98,289 tweets in Spanish was provided. Each tweet had a corresponding label representing the emoji that was in the tweet. The labels were assigned based on the frequency of the emoji, 0 being assigned to the most frequent emoji. The total number of labels was 20 for the English data and 19 for the Spanish data. We classified the tweets, our documents, by their emojis, our classes. We viewed the emojis as approximations of the sentiment expressed in the text.

For our baseline, we implemented a Bag of Words model using a Bernoulli Naive Bayes classifier. We analyzed the results and used the insights to implement our final system, which used a Linear Support Vector machine for classification and also used a Bag of Words model to represent each document. This system performed better than our baseline by ~3.5 percentage points for our English data and ~1.5 percentage points for our Spanish data. It also performed better than several neural network models we experimented with. Our macro F1 score were 32.73% and 17.98% for our English data and Spanish data respectively.

## 2 Baseline

For our Baseline, we used a Bag of Words model (BOW) with a Bernoulli Naive Bayes Classifier. We also implemented a Most Frequent Class Model (MFC) and a Random Model (RAND) to help draw insights from our baseline. The results of these models for the English and Spanish data are shown in Table 1 and Table 2 respectively. The tables show mean and standard deviation over the folds using 10-fold cross-validation. We reserved 10% of the data for testing and trained on the remaining 90% for each fold. We followed the same approach in all our experiments. The Micro F1 scores are heavily influenced by the performance of the dominant classes. Since ~21% of the tweets belong to label 0 (❤ ) for English, the micro F1 score was ~21% for the MFC model. Macro F1 scores, on the other hand, give equal weight to each class, since they average the F1 scores of each class.

We chose Bernoulli style Naive Bayes because it generally works better for short texts (e.g. Tweets) than its Multinomial counterpart (Manning et al., 2008). We empirically verified this with our task and data. To implement this model, we used the NLTK library for preprocessing and the scikit-learn framework for the model training (Bird et al., 2009; Pedregosa et al., 2011).

Our data pipeline consisted of four steps: Tok-

|  | Macro F1 |  | Micro F1 |  |
|---|---|---|---|---|
|  | Mean | S-Dev | Mean | S-Dev |
| **BOW** | 29.1 | 0.2 | 42.1 | 0.3 |
| **MFC** | 1.8 | 0.0 | 21.7 | 0.2 |
| **RAND** | 4.5 | 0.1 | 5.0 | 0.1 |

Table 1: Baseline results for English

|  | Macro F1 |  | Micro F1 |  |
|---|---|---|---|---|
|  | Mean | S-Dev | Mean | S-Dev |
| **BOW** | 16.5 | 0.4 | 29.6 | 0.4 |
| **MFC** | 1.7 | 0.0 | 20.0 | 0.4 |
| **RAND** | 4.8 | 0.2 | 5.5 | 0.2 |

Table 2: Baseline results for Spanish

|  | ❤️ | 😍 | 😂 | 😊 | 😎 | Total |
|---|---|---|---|---|---|---|
| ❤️ | 8637 | 618 |  |  |  | 10752 |
| 😍 |  | 2358 | 832 |  |  | 5145 |
| 😂 |  | 1022 | 2718 |  |  | 5114 |
| 😊 |  | 653 | 437 | 251 | 139 | 2313 |
| 😎 |  | 558 | 385 |  | 332 | 2092 |

Table 3: BOW Confusion Matrix for English. We have choose to leave some numbers out to prevent distraction from the patterns we are trying to show. All left out numbers are negligible.

enization, Bagination, Tf-idf transform and Training. For tokenization we used NLTK's Tweet-Tokenizer function. We configured it to convert all text to lowercase, crop repeating characters to a max of 3 and remove tweeter handles. We created the BOW by making a set of the tokens which appeared per the document. The next step was to normalize the frequency data into term-frequency times inverse document-frequency(tf-idf) representation. Finally, we trained a Bernoulli Naive Bayes classifier on this data.

## 2.1 Insights from BOW English Results

For the English tweets, the difference between the macro and micro F-scores for our Bernoulli model lies in the fact that the distribution of classes is very unbalanced. This is demonstrated from the results shown in Table 1.

Table 3 shows a confusion matrix for some labels of our BOW results for the English data. A mapping of select labels to emojis and their descriptions is shown in Table 4. The matrix illustrates the struggles of the BOW model. Generally many tweets are misclassified as 😍 and 😂 . Take 😎 for instance. While only 332 tweets were correctly labeled, 558 tweets and 385 tweets that should have been labeled 😎 were labeled 😍 and 😂 respectively. This misclassification of 😎 as 😍 and 😂 is not only the case for the selected classes represented in the matrix but also for most of the classes. Emojis 😊 and 😍 illustrate another pattern we observed. 653 out of 2312 tweets that should have been classified as 😊 were misclassified as 😍 . Since 😊 is a smiling face with smiling eyes and 😍 is smiling face with heart eyes, we inferred that the model might have trouble discriminating between classes with similar semantics.

That inference is supported by the fact that the model also struggles to discriminate between 📷 , a camera, and 📸 , a camera with flash. 458 tweets out of 1344 tweets (34%) that should have been labeled as 📸 where incorrectly labeled as 📷 . This is more significant when one notes that only 148 of 📸 tweets (11%) where labeled correctly.

To measure the impact of these semantically similar classes on our model, we collapsed classes that were semantically similar and reran the BOW model. All of the "heart" emojis were put into a single class; the "smiling" emojis (except the hearty eyes one) were put into another class; the "camera" emojis were collapsed into one class; and each remaining emoji had its own class. In the end, we had 13 classes after merging.

After running the model on these new set of classes, the macro F1 score improved by 6 percent points suggesting that the semantic similarity between emojis, such as 📷 and 📸 , has an effect although the effect was not nearly as significant as we had expected. It is worth noting that after collapsing the semantically similar classes, plenty of tweets were misclassified as the most frequent class. Thus, it may be that the semantic similarity of the classes does really matter but the gain in performance from collapsing the classes was offset by the fact that we had a class that was relatively much larger than the largest class before.

The BOW performed relatively well for labels 0, 1, 2, 4 and 17 (❤️ , 😍 , 😂 , 🔥 and 🎄 respectively) for the English data.

## 2.2 Insights from BOW Spanish Results

The Spanish results were much worse than the English results. Table 5 illustrates the major trend we noticed with the BOW model's performance on the Spanish data. Count is the number of tweets that correctly belong to a class. %C is what per-

| Label | Emoji | Description |
|---|---|---|
| 0 | ❤️ | Red heart |
| 1 | 😍 | Smiley face with heart eyes |
| 2 | 😂 | Face with tears of joy |
| 4 | 🔥 | Fire |
| 5 | 😊 | Smiling face with smiling eyes |
| 6 | 😎 | Smiling face with sunglasses |
| 13 | 💜 | Purple Heart |
| 10 | 📷 | Camera |
| 17 | 🎄 | Christmas Tree |
| 18 | 📸 | Camera with Flash |

Table 4: Some English Emojis

| | %**C** | %❤️ | %😍 | %😂 | **Count** |
|---|---|---|---|---|---|
| ❤️ | 62 | - | 14 | 0 | 1882 |
| 😍 | 43 | 16 | - | 14 | 1338 |
| 😂 | 46 | 0 | 25 | - | 908 |
| 💕 | 15 | 32 | 24 | 0 | 641 |
| 😊 | 9 | 14 | 32 | 17 | 647 |
| 😘 | 11 | 22 | 29 | 11 | 438 |
| 🇪🇸 | 63 | 5 | 17 | 4 | 331 |
| 😎 | 7 | 12 | 30 | 20 | 332 |
| 🎶 | 24 | 13 | 19 | 27 | 283 |

Table 5: BOW Spanish results. %**C** refers to the percent correctly labeled and %*emoji* refers to the percent mislabeled as *emoji*.

centage of tweets were correctly labeled. %❤️ , %😍 and %😂 are the percentages of tweets that our model misclassified as labels 0, 1 and 2 respectively. Thus for ❤️ , there were 1882 tweets in the test data; 62% of these were labeled correctly while 14% was incorrectly as 😍 . As the table shows, a significant percentage of tweets were either misclassified as ❤️ , 😍 or 😂 . The exception to this is 🇪🇸 - the Spanish flag. Table 6 shows the emojis corresponding to the numerical labels.

## 2.3 Neural Network

Upon reading about how neural models achieved high scores on similar tasks, we decided to try out methods based on (Barbieri et al., 2017) and (dos Santos and Gatti, 2014). The task this paper tries to solve is based on Barbieri et al.'s work where they do the same task. Their best performing model was a character based Bi-directional Long

| Label | Emoji | Description |
|---|---|---|
| 0 | ❤️ | Red heart |
| 1 | 😍 | Smiley with heart eyes |
| 2 | 😂 | Face with tears of joy |
| 3 | 💕 | Two hearts |
| 4 | 😊 | Smiley with smiling eyes |
| 5 | 😘 | Face blowing a kiss |
| 9 | 🇪🇸 | Spain |
| 10 | 😎 | Smiling face with sunglasses |
| 16 | 🎶 | Musical notes |

Table 6: Some Spanish Emojis

Short-term Memory Network (char-BLSTM). We also took inspiration from dos Santos and Gatti's work. They were able to achieve very good results using a Convolutional Neural Network (CNN) to do sentiment analysis. We tested four different types of neural network models: LSTM, BLSTM, CNN-LSTM and CNN-BLSTM. For the LSTM based models, we used a network size of 128. The only difference between our LSTM and our BLSTM is that we added a layer to train each input bidirectionally. Our CNN's convolution layer had an output dimension of 64 and a kernel size of 5. For it's pooling layer we chose a pool size of 4. When training each of the neural network models we used a development set which was 10% of the training set to select the best parameters and to know how many epochs to train for. We settled on these specific parameters after trying out different parameters on the development set. None of our neural network models performed significantly better than our baseline.

## 2.4 Linear SVM

Realizing that our neural network models did not perform any better than our BOW baseline, we decided to try a BOW model with other classifiers which were not neural networks. We settled on a Linear Support Vector Machine. To enable multi-class classification we used a one-vs-rest approach.

## 2.5 Sampling

Roughly 20% of the tweets in the English data belong to label 0. The performance of classifiers such as Naive Bayes degrades when there is such a

|  | English |  | Spanish |  |
|---|---|---|---|---|
|  | **Mean** | **S-Dev** | **Mean** | **S-Dev** |
| **Base** | 29.10 | 0.20 | 16.49 | 0.42 |
| **C+L** | 29.30 | 0.36 | - | - |
| **C+L(f)** | 29.35 | 0.44 | - | - |
| **LSVM** | 32.73 | 0.24 | 17.98 | 0.31 |

Table 7: Macro F1 scores. Base is baseline, L is LSTM, C is CNN, (f) means each class was equally represented.

dominant class (Rennie et al., 2003). This data imbalance exists in the Spanish data as well. To improve the performance of our classifiers, we perform a sampling of the data so that we train on a data set where the classes are roughly equally represented. We performed a simple under sampling by randomly selecting an equal number of tweets from each class even though a more sophisticated re-sampling method will likely improve the results (Estabrooks et al., 2004).

## 3 Results

The neural network models that we tested ended up achieving around the same score as our BOW baseline. The BOW model with a Linear Support Vector Machine for classification provided the best results. Table 7 shows the results of the LSVM along with the results of our baseline and our best performing neural network models for comparison. The effect of sampling the dataset to balance the frequency of each class was negligible as shown in Table 7. The improvement to employing sampling was 0.05 percentage points for our CNN combined with LSTM model. The F1 score of our LSVM model on the test data from the task organizers was 31.834 which is within one percent of the 32.73 from our 10-fold cross-validation. Precision on the test data was 39.803, recall was 31.365 and accuracy was 45.732. [1]

## 4 Discussion

The first important trend we observe with our system (BOW model with LSVM classifier) is the most frequently seen emojis ❤ , 😍 and 😂 perform well in terms of true positives - ~83% for ❤ (8848/10622), ~57% for 😍 (2903/5077) and ~63% for 😂 (3171/5067) while at the same time being false positives for many classes. Take 😊 (label 5) for instance. 327 tweets are correctly classified as belonging to 😊 . However, 732 tweets that should have been classified as 😊 are misclassified as 😍 . The trend of misclassifying more tweets as 😍 was seen for labels 6, 7, 8, 13, 14, 15, 16 and 19 as well. This trend carried through from our baseline; the final system performs better only because of marginal improvements in the classification itself.

Below are some tweets for 📸 that the LSVM succeed in classifying that the Bernoulli Naive Bayes could not find. We choose 📸 because the percentage difference in performance (in favor of the LSVM) is the greatest here.

> Different angles to the same goal. by @user @ New
>
> When iris.apfel speaks...knowledge and wisdom is all you hear so listen up... :@drummondphotog

Our supposition is that the Linear Support Vector Machine is able to make associations that the Bernoulli Naive Bayes is unable to make. ”Angles”, we suspect, correlates with camera than the other emojis and the LSVM finds that association. The second tweet is interesting because it would seem that the LSVM is able to connect the “photog” in “@drummondphotog” despite our use of a Bag of Words Model unless the prediction was based on some other less obvious word in the tweet.

## Acknowledgments

This project was carried out as a part of CS 8761, Natural Language Processing, offered in Fall 2017 at the University of Minnesota, Duluth by Dr. Ted Pedersen. We are grateful to Dr. Ted Pedersen for his support and guidance in this project. Authors are listed in alphabetical order.

## 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

[1]Task Scoreboard https://docs.google.com/spreadsheets/d/1on1Oj53EFcE4n-yO_sJc1JEo6x8hcUh5hsWTkTYdc_o/edit#gid=885431079. We submitted under theteam name: Hopper.

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.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media.

Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. 2004. A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20(1):18–36.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schuetze. 2008. *Introduction to Information Retrieval*. Cambridge University Press. Pg. 268.

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.

Jason D. M. Rennie, Lawrence Shih, Jaime Teevan, and David R. Karger. 2003. Tackling the poor assumptions of naive bayes text classifiers. In *In Proceedings of the Twentieth International Conference on Machine Learning*, pages 616–623.

Cicero dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.