Emoji Embeddings Across Languages and Cultures

Alex Dennis, Christian Muxica, Sam Dziewietin, Simon Koch-Sultan
{acdennis,cmuxica,sdziewietin,skochsultan}@umass.edu

> ☎️👨🏻⛵🐳👌🏻
> Call me Ishmael.
>
> *Emoji Dick*[1]
> Herman Melville
> ed. Fred Benenson

1 Introduction/Problem Statement

Emoji have become an integral part of online communication. Emoji use has risen as older Internet slang has decreased in usage. Nearly half of all text on Instagram contains emoji (Dimson, 2015) and five billion emoji per day are sent over Facebook Messenger (Burge, 2017). In 2015, Oxford Dictionaries declared the Face with Tears of Joy (😂) the Word of the Year, reflecting the newly prominent role emoji play in written communication (Oxford Languages, 2015). Contemporary computational social science, communication research, and linguistic research have shown that communication through emoji has the same level of complexity and nuance as any other part of natural language.

Because of this rise in use, it has become increasingly important for researchers to incorporate emoji when working on natural language processing tasks. We hope to use natural language processing tools to aid in analysis of the meaning and use of emoji, especially as that use differs between language and culture. We expand on previous work by introducing state-of-the-art models to the task. We produced embeddings for emoji using ELMo in an attempt to improve on earlier work using word2vec. Furthermore, we used qualitative techniques to compare the cross-cultural meanings of emoji, specifically between English and Portuguese Twitter users.

2 What we proposed vs. What we accomplished

- ✓Scrape Twitter for tweets containing emoji
- ✓Create sets of tweets written in English and Portuguese
- *Create sets of tweets written in the United States and Brazil*: Due to limited information available in the scraped tweets, we were unable to reliably identify the location where a tweet was posted
- ✓Preprocess scraped tweets
- ✓Train word2vec models on tweets
- ✓Train ELMo models on tweets
- ✓Produce 2D projections of emoji embeddings
- *Align English with Portuguese embeddings for cross-linguistic comparison*: We were unable to implement embedding alignment in the time available
- *Develop prediction task to compare ELMo and word2vec performance*: We discovered that ELMo and word2vec are not directly quantitatively comparable

3 Related Work

Semiotics research has found that a large portion of the pragmatic purpose of emoji can be explained as replacing the loss of prosodic and gestural information that inherently occurs when using the written medium, and to generally convey emotion (often positive by default, especially for the depiction of ordinary objects) (Danesi, 2016).

Psycholinguistic work has confirmed that emoji can be a integral part of linguistic meaning at the processing level. Speakers will have the same

[1] http://www.emojidick.com/

brain responses associated with irony expressed in words when that irony is expressed using emoji (Weissman and Tanner, 2018). These processing patterns could be specific to language, but there is strong evidence that they could be domain-general, reflecting how our abilities to communicate through language can change with the advent of technological augmentation (Cohn, 2019).

Despite this potential to resolve some of the ambiguity of the textual medium, the meaning of emoji—much like prosodic and gestural features themselves—is still often open to interpretation and may lead to miscommunication, especially during cross-platform communication (Miller et al., 2016). While most emoji are used similarly across cultures, some (particularly symbols) exhibit some cross-cultural differences (Guntuku et al., 2019). For example, Twitter and Weibo users associate different sets of emoji with different emotions (Li et al., 2019).

Researchers have investigated this problem using word embeddings to measure the similarity of pairs of emoji. Some pairs of emoji are used in dissimilar contexts in Barcelona than in Madrid, suggesting different cultural associations (Barbieri et al., 2016a). These findings have extended to an investigation of emoji use across American and British English, Peninsular Spanish, and Italian (Barbieri et al., 2016b).

Handling the nuanced meanings expressed by emoji has recently become a topic of interest in research. Prasad et al. (2017) handled emoji while performing sentiment classification on tweets by manually creating a dictionary from emoji to either positive or negative sentiment. Kralj Novak et al. (2015) developed a lexicon of emoji sentiment based on annotated tweets.

Researchers have taken to using word2vec to interpret emoji. Training skip-gram word2vec models on tweets containing emoji produced vectors whose cosine similarities compared well to human ratings (Barbieri et al., 2016c). Training instead on the Unicode descriptions of emoji produced vectors that achieved state-of-the-art results in Twitter sentiment classification. The resulting set of embeddings, called emoji2vec, were specifically trained occupy the same vector space as the original Google News word2vec embeddings (Eisner et al., 2016; Mikolov et al., 2013). A strategy of replacing emoji with their Unicode description saw improvements in irony detection and sentiment analysis over emoji2vec (Singh et al., 2019).

One issue with word2vec is that the embedding for a given word type is always the same no matter the context. For example, in the sentences "I opened a new bank account" and "I sat by the river bank", both instances of the 'bank' token would have the same embedding in word2vec even thought they have very different meanings. In order to create contextualized word embeddings, Peters et al. (2018) used a multi-layer bidirectional LSTM to incorporate the context both before and after the given token.

4 Your Dataset

Some form of preprocessing is almost always involved in the creation of a corpus, especially when the source of the data is more naturalistic. As few NLP researchers have been interested in emoji historically, the removal of these characters is often a part of a standard preprocessing regimen. Additionally, NLP research has been dominated by the study of English and resources can be sparse for other languages. Thus, our needs for text data containing emoji in both English and Portuguese essentially ruled out any pre-existing corpora. The one avenue we did explore was the Twitter dataset used by Barbieri et al. (2016c). However, the authors informed us via email that they could not share this data on account of Twitter terms of agreement. Thus, we elected to construct our own corpus of bilingual Twitter data by mean of scraping. We began with the popular python library Tweepy[2], but switched to Twint[3] after discovering it allowed us to subvert the normal scraping maximums.

4.1 Twitter Corpus

The final corpus we constructed consists of 280,000 individual tweets—140,000 coming from each language. All tweets scraped were written between the years of 2017 and 2018. The language of a tweet was identified by virtue of the authors registered language on their Twitter profile. As well, our search was filtered such that no retweets were included. We scraped for tweets containing any one of a set of 16 different emoji: 😂, 😍, 😊, 🙏, 💕, 😭, 😢, 😔, 😡, 👏, 🔥, ⚽, 🏆, 🍀, 🍆, and 🍑. Each of these emoji was chosen for one of three reasons. First, each of these emoji was

[2] https://www.tweepy.org/

[3] https://github.com/twintproject/twint

at least within the 8th group of median frequency as reported by the Unicode Consortium[4] website. Second, each emoji was chosen to at least have one closely related pair which should logically cluster together in an embedding projection. For example, the heart (💕) and heart eyes (😍) emoji. Third, some emoji were picked for the potential to yield variation in use between languages. For example, the praying (🙏) and soccer ball (⚽) emoji. Soccer is far more popular outside of the United States and religious practices differ between Anglophone and Lusophone cultures.

4.2 Data Preprocessing

The bulk of prepossessing involved removing or cleaning idiosyncrasies of Twitter using regular expressions. All links and hashtags were removed. Links were taken out as they contain little actual relevant linguistic information and could negatively impact model performances. While hashtags often convey information relevant to the tweet, they do so in atypical fashion similar to emoji. Hashtags are interesting from an NLP perspective in their own right, but their inclusion would have been another manipulation beyond the scope of this study.

We elected to include references to other users introduced by the @ character. However, we replaced all of these references with an "[UNK-USER]" token. User @s often carry both important semantic and syntactic information on Twitter, often referring to an entity within a discourse. We felt compressing this information into an UNK token was a logical compromise between including all user references and removing them entirely. A similar compromise was made with regard to other emoji. The Twint scrapper we utilized could search Twitter for a list relevant characters, but it could not exclude characters from this search. As such, our tweets contained a substantial amount of other emoji which we were not interested in. A regular expression for extracting emoji written by Elias Dabbas [5] was adapted to detect emoji which did not belong to the relevant group. All of these emoji were replaced with an "[UNK-EMOJI]" token as well. Removing these emoji felt excessive, as all emoji contribute some meaning to a sentence. However, keeping all emoji would have been beyond the scope for this project. Again we compromised with an "[UNK-EMOJI]" token in the hope that it would encode some information which would benefit our model.

Tokenization was handled almost entirely by spaCy—the only exceptions being emoji. As spaCy is not designed with emoji in mind, spaces were added between emoji and all other characters. Otherwise spaCy would tokenize emoji together with each other and with other words. This would have erroneously generated embeddings such as "💕😍" and "😭[UNK-EMOJI]" in our models.

Before inputting the cleaned dataset into our models for training, we split our corpora into three sections for each language. These three divisions corresponded to a training, validation, and test set. The sections accounted for 60%, 20%, and 20% of the total amount of tweets in each language respectively. Numerically this comes out to 84,000/28,000/28,000 tweets in the splits for both English and Portuguese.

4.3 Baseline Dataset

In addition to our Twitter dataset, we also trained word2vec on two larger general corpora, one for each language. These baseline data sets consist of text from Wikipedia obtained from Linguatools' Wikipedia Monolingual Corpora[6]. The article contents were extracted from the XML files with the xml2txt Perl script available at the same webpage. We discarded any content enclosed in math and table tags, as well as on any disambiguation pages. This was intended to restrict the content of the corpora to natural language. Once we obtained the cleaned data set, we appended the cleaned tweets to obtain the combined data set used to train word2vec.

While there were no major issues in training a model on the full contents of the Portuguese Wikipedia, we encountered issues with using the full English Wikipedia. The cleaned Portuguese Wikipedia file was 1.8GB in size and contained 319,844,644 tokens. The English file was 13GB in size and contained 2,319,783,831 tokens. When training a word2vec model on the full English Wikipedia, the loss dropped to 0 by the second epoch and training took over seven hours to run compared to 40 minutes for Portuguese. We de-

[4] https://home.unicode.org/emoji/emoji-frequency/

[5] https://www.kaggle.com/eliasdabbas/how-to-create-a-regex-to-extract-emoji

[6] https://linguatools.org/tools/corpora/wikipedia-monolingual-corpora/

cided to use only the first 10 million lines from the cleaned text file for a 1.7GB file and then append the tweet test set.

5 Baselines

We chose to use word2vec as the baseline against which to compare ELMo. This portion of our project involved essentially replicating emoji2vec in that we generated embeddings for emoji. Since Eisner et al. (2016) used word2vec to generate their embeddings, we decided to do the same for comparison.

We implemented our word2vec model using Gensim (Řehůřek and Sojka, 2010). Our 300-dimensional models were trained across 5 epochs as CBOW models using the mean of the context word vectors. For negative sampling, we used the Gensim default values of 5 "noise" words to be drawn and an exponent of 0.75 to shape the negative sampling distribution, as used in the original word2vec paper (Mikolov et al., 2013). We used an alpha value of 0.025, decreasing linearly to 0.0001 during training, also per the Gensim default. Since word2vec does not use validation or testing, we trained using the same 60% training data subset. For training performance, see Figure 1.

The results of our word2vec models in comparison to ELMo are detailed in section 7.1.

5.1 Unused Baselines

There were other baselines that had attractive qualities that we ultimately declined to use for this project. For example, GLoVe (Pennington et al., 2014) was attractive because it was originally trained on Twitter data, but it lacked the Portuguese embeddings that would have been necessary for our project.

We also considered setting the bar higher for ourselves with more recent embeddings. FastText's (Joulin et al., 2018; Bojanowski et al., 2017) aligned pre-trained embeddings had several advantages. Their method of alignment helped mitigate the "hubness" problem of particular embeddings being in close proximity to a disproportionate number of other embeddings without degrading the quality of the loss function. However, the lack of emoji in their dataset ruled it out as an alternative.

Ultimately, word2vec's prior ubiquity won out for us as a baseline option.

6 Your Approach

For our state-of-the-art model, we used ELMo (Peters et al., 2018). We trained ELMo on our twitter data starting from a pre-trained model. We used the 5.5B English model and the Portuguese model from the AllenNLP website[7]. The English 5.5B model was trained on 5.5 billion tokens from Wikipedia and news sources. The Portuguese was trained on text from Wikipedia.

We used the AllenNLP library to implement and train the ELMo model. The library has an implementation of the ELMo model, but it is not designed for our task; instead, it is designed for fine-tuning the model over a down-stream task. Therefore, we had to design our own task/loss based on parts of the built-in implementation (specifically, a linear prediction layer combined with a cross-entropy loss).

First, we tokenized the text using the spaCy[8] tokenizer (for English and Portuguese). Then, these tokens were indexed in two different ways: a character-based indexing for input into ELMo, and a token-based indexing for use in our prediction task. Next, the character-based indexed tokens are fed into ELMo to produce contextualized representations. Finally, these contextualized representations are fed into a linear prediction layer to which softmax is applied to get the prediction probabilities. We minimized the cross entropy loss in both the forward and backward directions (we designed the loss to be as close as possible as what was originally used in Peters et al. (2018))

$$\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K_n}(-\log p(x_{nk}|x_{n1},\ldots,x_{n(k-1)})) \\ +(-\log p(x_{nk}|x_{n(k+1)},\ldots,x_{nK_n}))$$

where N is the number of data instances (i.e., individual tweets), K_n is the length of the n^{th} data instance, and x^{nk} is the k^{th} token.

We trained[9] the model using the Adam optimizer (with a learning rate of 0.001) over 10 epochs (with each epoch running through the entirety of the data once). To prevent over-fitting, we evaluate the model on validation data each epoch and keep the model with the best validation loss. We ran the training process on Google Colab. Because of the memory limitation of the Colab GPU,

[7] https://allennlp.org/elmo

[8] https://spacy.io/

[9] all code for training ELMo are contained in the elmo_train.ipynb notebook

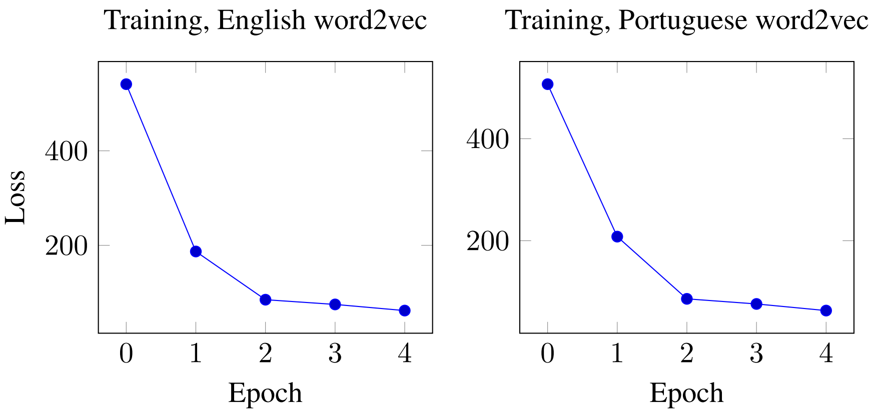


Figure 1: Training loss per epoch for word2vec.

we had to limit the maximum length of a data instance to 100 tokens and limit the maximum vocabulary size to 100,000 tokens.

In addition to training ELMo fine-tuned on our dataset, we also include an non-fine-tuned version of ELMo. This model was achieved by training the model as usual on our dataset, with the ELMo portion of the model frozen (i.e., it will retain its parameters from the pre-trained model). We will call this the frozen model. We include the frozen model as a control to compare against in determining whether our fine-tuned ELMo is learning useful representations.

In evaluating the models, we use two kinds of performance metrics, loss and combined perplexity. The combined perplexity is the geometric mean of the perplexity of the forward sub-model and the perplexity of the backward sub-model

$$\exp\Big(\big(\frac{1}{N}\sum_{n=1}^{N}\frac{1}{K_n}\sum_{k=1}^{K_n} \frac{1}{2}(-\log p(x_{nk}|x_{n1},\ldots,x_{n(k-1)}) + (-\log p(x_{nk}|x_{n(k+1)},\ldots,x_{nK_n})))$$

At each training epoch, we measure the loss and combined perplexity on both the training and validation datasets. See Figure 2 for training performance.

7 Results

7.1 Embedding projections

We generated a variety of embedding projections using the online TensorFlow projector[10], six of which are displayed in Figure 3. The ELMo embeddings utilized in these projections are an average across the second layer of contextualized embeddings for every sentence in the test data. We chose the second layer to maximize the likelihood of representing semantic information and not just the obvious syntactic difference between words and emoji. We chose an additional set of 16 attested words for each language to project alongside our 16 emoji. These words were chosen such that each had a strong semantic relation to at least one of the the emoji. As well, these words came from a variety of parts of speech. These choices were made to provide multiple points of comparison with which to evaluate the information encoded in the emoji embeddings.

All of these 2-dimensional projections were generated using the t-Distributed Stochastic Neighbor Embedding or t-SNE dimensionality reduction algorithm (Maaten and Hinton, 2008). Each projection was run to 5,000 iterations with a perplexity setting of 5, learning rate of 10, and a single perturbation at 2,500 iterations before being downloaded and labelled. The high iteration was chosen to ensure that the projection had settled into a steady state before interpreting. The perplexity hyper parameter correlates (roughly) with the balance of projecting local and global effects present in the data[11]. At a low perplexity the clustering projected reflects more local variations between tensors and at a high perplexity more global variation. It is also recommended that the perplexity value not exceed the number of points being modeled. Thus, we settled on a value of 5 for perplexity in the hope that the projections would

[10] https://projector.tensorflow.org/

[11] https://distill.pub/2016/misread-tsne/

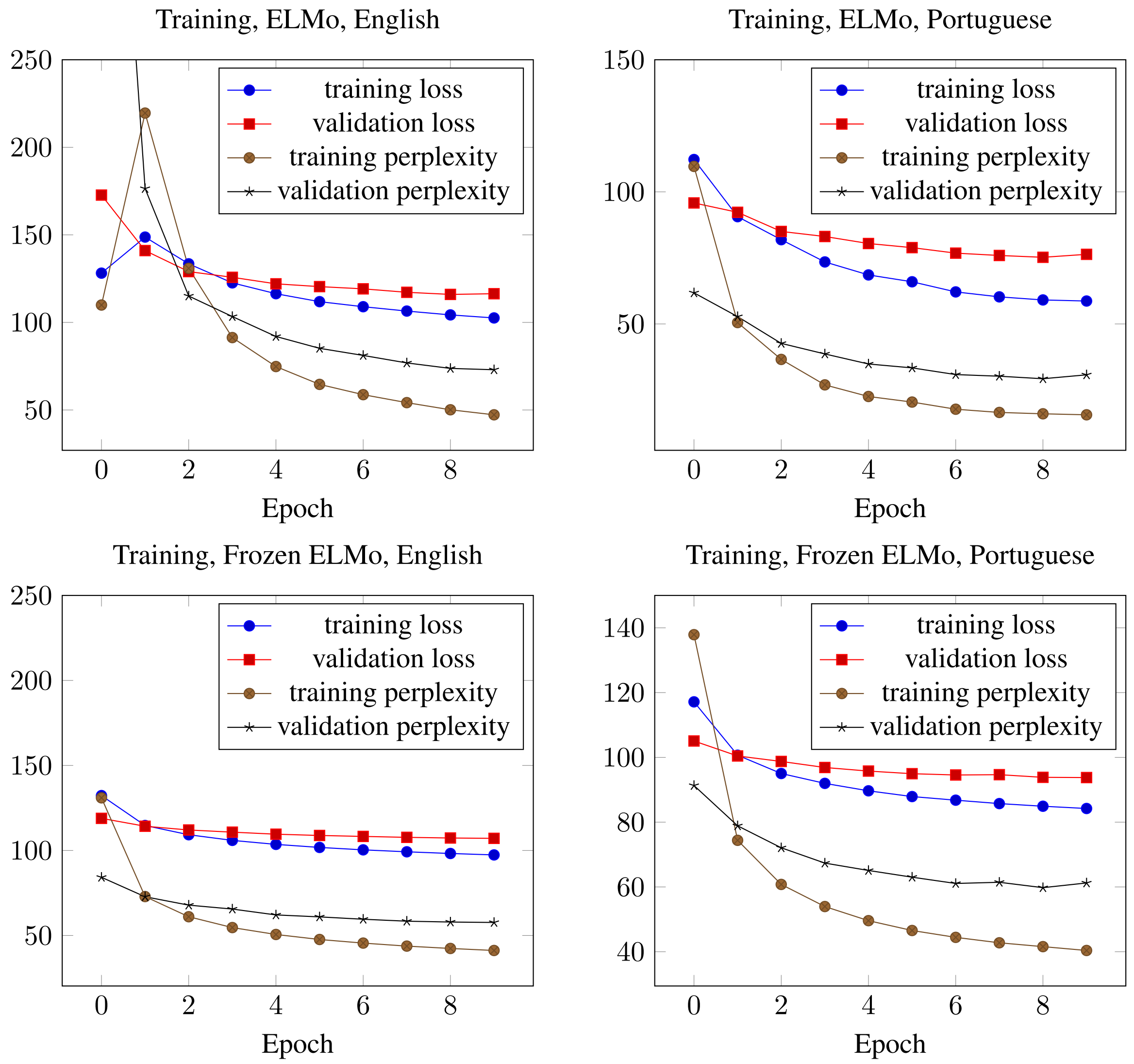


Figure 2: Metrics per Epoch while training ELMo

capture both local and global effects. The learning rate of 10 was chosen on the basis that higher rates never seemed to stabilize and slower learning rates simply seemed to take longer to stabilize.

The most immediate trend across all of these projections is the clear split between words and emoji. This split is more extreme in some cases then others, specifically figures 3d and 3f, but it is present in all regardless. This speaks to the fact that emoji are used in a fashion which is distinct from words—at least the ones included in the projections. We originally expect that emoji would overlap at least somewhat with words, but the strong emphasis on syntax appears to still remain after two layers. Within the emoji themselves, however, it is clear that some level of meaning is still being encoded since in that case collocations themselves inherently suggest a relation between the two characters. This effect is not absolute, since when making more complex utterances out of emoji, people will tend to collocate characters that have situation-specific (but not inherent) associations, but inherent associations should be prevalent enough that they should not be clouded out by others.

We remain confused by the difference in cluster shape between the projections in figures 3d and 3c. In figure 3d the emoji and word clusters are much farther apart and much tighter. We have run the t-SNE algorithm on the fine tuned Portuguese embeddings multiple times to account for stochastic factors—the output remains relatively the same. This may speak to some difference between the English and Portuguese data we fine-tuned on. This is supported by the fact that the shape of the frozen English ELMo embeddings is similar in character.

Lastly, word2vec generates quite similar projections as compared to fine-tuned ELMo in both languages despite the fact that ELMo embeddings outperform those of word2vec. One reason for this might be that we have averaged across all of the contextual ELMo embeddings for the test set. The key advantage of ELMo over previous models is the way in which it contextualizes meaning. Averaging across contexts likely obfuscates the quality of ELMo embeddings over those of word2vec. Unfortunately this style of projection might simply be an ineffective way to evaluate the performance of ELMo.

7.2 Perplexity

We evaluated the ability of our fine-tuned ELMo model to learn the contextual representations of the emoji by comparing against the non-fine-tuned frozen ELMo model. We compared the combined perplexity between both of these models for each language. All values are computed over the test dataset.

	English	Portuguese
Frozen	57.4779	60.1325
Fine-tuned	73.9264	28.3435

Table 1: Combined Perplexity (lower is better)

For English, we see that the frozen model actually performed better than the fine-tuned model. We suspect that this is because the pre-trained model already contained enough information to perform well on our test data, and because ELMo was frozen, was able to focus on the prediction layer. For Portuguese, we see that the fine-tuned model performed better than the frozen model. This makes sense, since the pre-trained Portuguese was trained on Wikipedia text, which is a more formal register compared the naturalistic language found in tweets.

These perplexity values are computed over every token in the test data set, which may not be the best metric of performance over emoji specifically. Therefore, we also computed this perplexity over only the emoji tokens. See table 2.

Over just the emoji tokens, the fine-tuned model performed better than the frozen model. As before, the difference in performance was much greater in the Portuguese model. Again, we suspect this is due to a difference in the training data available to the pre-trained models.

	English	Portuguese
Frozen	27.7369	24.8520
Fine-tuned	25.6278	13.4775

Table 2: Combined Perplexity over Emoji Tokens (lower is better)

8 Contributions of Group Members

- **Alex**: ELMo training and quantitative evaluation
- **Christian**: scraping, preprocessing, interpretation
- **Sam**: baseline dataset, word2vec training, projections
- **Simon**: dataset protocols, projections, baseline research, interpretation

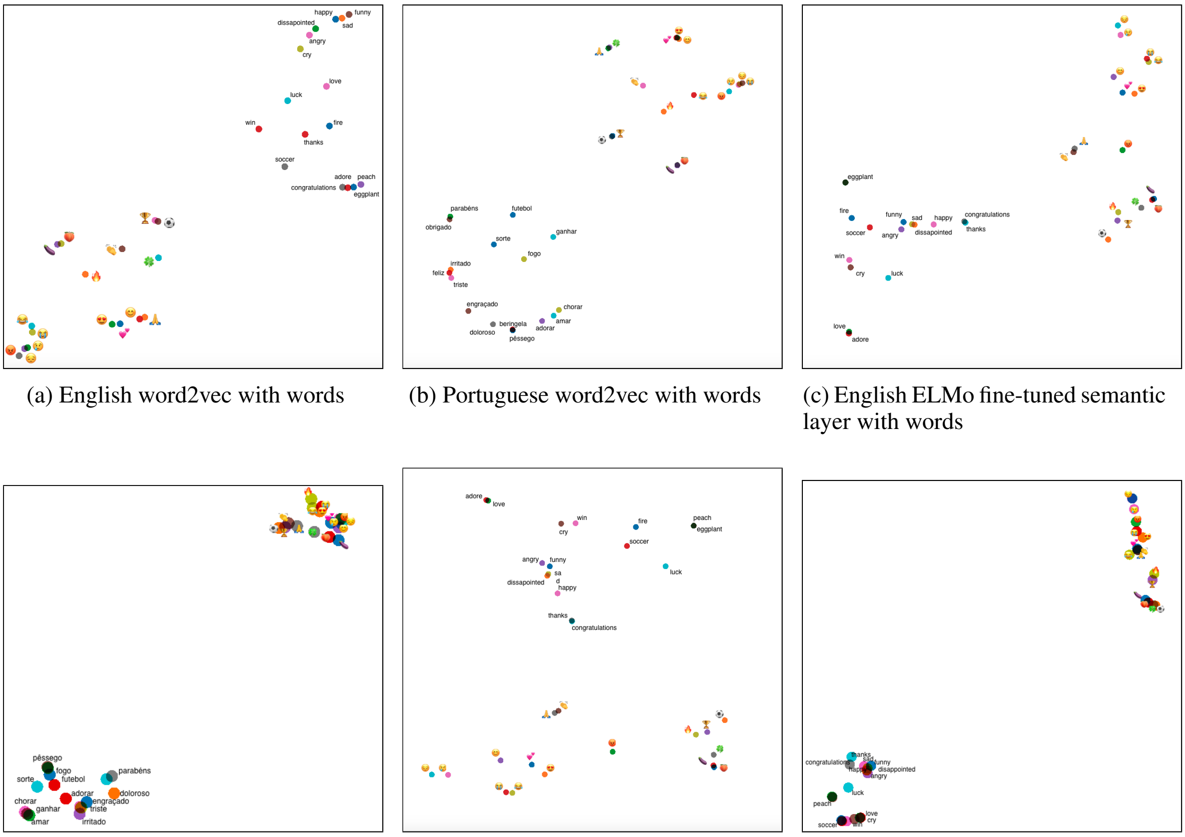
9 Conclusion

Working with emoji led to some unexpected complications. We were not able to use certain emoji in our subset because of multi-codepoint emoji.

The projections also proved more difficult than we were originally expecting. For one thing, ELMo embeddings are contextual and have to be averaged to get point representations that can be fed into t-SNE. A more accurate analysis would involve a probability distribution for each word type. Unfortunately, this kind of visualization was beyond the scope of our project. Additionally, the tokenized data from spaCy were not lemmatized, so we could only get masculine singular forms of Portuguese adjectives. A more morphologically rich language being thrown in the mix likely would have presented even more of a problem.

The apparent stubborn robustness of the syntactic nature of the projections, even in the purportedly semantic layer of ELMo, was a bit of a surprise. Emoji and words were generally pretty firmly clustered in separate groups.

We did notice some cultural characteristics of emoji that were of note. For example, the clover emoji (🍀) embedding was near the eggplant (🍆) and peach (🍑) in almost every projection, suggesting a universality to the innuendo of "getting lucky." We also discovered that the crying emoji (😭) was more likely to be associated with negative emotions in the Portuguese data, and with positive emotions in the English data.



(a) English word2vec with words

(b) Portuguese word2vec with words

(c) English ELMo fine-tuned semantic layer with words

(d) Portuguese ELMo fine-tuned semantic layer with words

(e) English ELMo fine-tuned contextless layer with words

(f) English ELMo frozen semantic layer with words

Figure 3: Embedding Projects for various models

9.1 Limitations and Future Work

An expanded version of this project would likely first and foremost generate embeddings for all emoji, compare more than two languages to see which languages' emoji representations were the most similar to each other, and account for geographic data to highlight diversity within populations that speak the same language. We would also like to create different embeddings with sensitivity to finer-grained intralinguistic features of tweets such as socioeconomic status, political affiliation, age, etc., to the degree that this information can be discovered or estimated.

Variation in emoji use could also be investigated across modalities. This project has only utilized data scraped from Twitter, but there are various other possibilities. Particularly the use of emoji within any kind of direct messaging format. A private message between two individuals is a very different linguistic context from any kind of social media micro-blog. Not only would this comparison be interesting, but it could prove useful in developing chat bots capable of utilizing emoji more naturally in a one on one discourse.

We also wanted to include formal linguistic components into our work, and did not really find much room to do so. As semantics and pragmatics advance with respect to describing various forms of human communication, we would like to test the ability of emoji embeddings to represent new categorizations invented for emoji (e.g. as discourse particles) in some kind of classification probe task.

A potentially fruitful extension of this work would be to test the extent of the effect of emoji to replace prosodic elements found in spoken language. This could be achieved through alignment of word embeddings generated from Twitter containing emoji data with word embeddings generated from corpora that contain prosodic information. There would likely be a considerable difference in register between the two corpora, but if emoji and corresponding prosodic elements can be found somehow to play similar roles, it would be an excellent way to demonstrate the level of sophistication of encoding inherent in emoji embeddings and prosodic marking embeddings. This would have implications for directions in theoretical linguistics and psycholinguistics, with associated emoji characterizing the role of particular prosodic patterns and vice versa. Of course, underlying all of this would be a highly robust system using contextualized word embeddings. Prosodic markers would naturally need embeddings all their own, and not the same subword breakdown that one might find with BERT, for example.

It would also be nice to test the impact of distance from the end of the tweet of various emoji on their contextual embeddings, since this appears to affect the strength of the sentiment being expressed in the semiotics literature. Given the importance of emoji placement to the sentiment and flow of a tweet, it would also be useful to fit emoji into an understanding of Internet language syntax, if any present or future formalism is compatible.

We would also want to use a model with more layers, such as BERT, which would actually more seriously show semantic characteristics in the embeddings generated at the higher layers. The robust subword tokenization would also allow us to analyze novel edge cases composed of alphanumeric characters, emoji, and punctuation in a more intelligent way.

We would like to incorporate cross-linguistic word embeddings, as it would allow more direct cross-linguistic comparison. This is because the embeddings for both languages would be in the same geometric space. For example, we would be able to take average embeddings from each language and compare their distance.

9.2 Final Thoughts

Language is not a static artifact, but rather a fluid and nuanced cognitive and social mechanism worthy of examination. In the present internet age, we are more empowered than ever before to study the linguistic behaviors that emerge in various communities. We think that natural language processing provides valuable tools to analyze and understand language, and we hope to see these tools further develop for the benefit of humanity.

References

Barbieri, F., Espinosa-Anke, L., and Saggion, H. (2016a). Revealing patterns of twitter emoji usage in barcelona and madrid. *Frontiers in Artificial Intelligence and Applications. 2016;(Artificial Intelligence Research and Development) 288: 239-44.*

Barbieri, F., Kruszewski, G., Ronzano, F., and Saggion, H. (2016b). How cosmopolitan are emojis?: Exploring emojis usage and meaning over different languages with distributional semantics. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 531–535. ACM.

Barbieri, F., Ronzano, F., and Saggion, H. (2016c). 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)*, pages 3967–3972.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Burge, J. (2017). *5 Billion Emojis Sent Daily on Messenger*. https://blog.emojipedia.org/5-billion-emojis-sent-daily-on-messenger/.

Cohn, N. (2019). Your brain on comics: A cognitive model of visual narrative comprehension. *Topics in Cognitive Science*.

Danesi, M. (2016). *The Semiotics of Emoji: The Rise of Visual Language in the Age of the Internet*. Bloomsbury Academic, UK.

Dimson, T. (2015). *Emojineering Part 1: Machine Learning for Emoji Trends*. https://instagram-engineering.com/emojineering-part-1-machine-learning-for-emoji-trendsmachine-learning-for-emoji-trends-7f5f9cb979ad.

Eisner, B., Rocktäschel, T., Augenstein, I., Bošnjak, M., and Riedel, S. (2016). emoji2vec: Learning emoji representations from their description. *arXiv preprint arXiv:1609.08359*.

Guntuku, S. C., Li, M., Tay, L., and Ungar, L. H. (2019). Studying cultural differences in emoji usage across the east and the west. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 13, pages 226–235.

Joulin, A., Bojanowski, P., Mikolov, T., Jégou, H., and Grave, E. (2018). Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Kralj Novak, P., Smailović, J., Sluban, B., and Mozetič, I. (2015). Sentiment of emojis. *PLOS ONE*, 10(12):1–22.

Li, M., Guntuku, S., Jakhetiya, V., and Ungar, L. (2019). Exploring (dis-) similarities in emoji-emotion association on twitter and weibo. In *Companion Proceedings of The 2019 World Wide Web Conference*, pages 461–467. ACM.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Miller, H. J., Thebault-Spieker, J., Chang, S., Johnson, I., Terveen, L., and Hecht, B. (2016). “blissfully happy” or “ready tofight”: Varying interpretations of emoji. In *Tenth International AAAI Conference on Web and Social Media*.

Oxford Languages (2015). *Word of the Year 2015*. https://languages.oup.com/word-of-the-year/2015/.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations.

Prasad, A. G., Sanjana, S., Bhat, S. M., and Harish, B. (2017). Sentiment analysis for sarcasm detection on streaming short text data. In *2017 2nd International Conference on Knowledge Engineering and Applications (ICKEA)*, pages 1–5. IEEE.

Řehůřek, R. and Sojka, P. (2010). Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. http://is.muni.cz/publication/884893/en.

Singh, A., Blanco, E., and Jin, W. (2019). Incorporating emoji descriptions improves tweet classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2096–2101.

Weissman, B. and Tanner, D. (2018). A strong wink between verbal and emoji-based irony: How the brain processes ironic emojis during language comprehension. *PLOS ONE*, 13(8):1–26.