Acquiring Cross-Domain Representations for Contextual Detection Using Extensive Emoji Data

Abstract

This research delves into the application of a vast collection of emoji occurrences to acquire versatile representations applicable to diverse domains for the purpose of identifying sentiment, emotion, and sarcasm. Natural Language Processing (NLP) tasks frequently encounter limitations due to the deficiency of manually labeled data. In the realm of social media sentiment analysis and associated tasks, researchers have thus employed binarized emoticons and specific hashtags as a means of distant supervision. Our study demonstrates that by broadening distant supervision to include a more varied array of noisy labels, models can achieve richer representations. Through emoji prediction on a dataset encompassing 1,246 million tweets, each including one of 64 prevalent emojis, we achieve state-of-the-art results on eight benchmark datasets focusing on sentiment, emotion, and sarcasm detection, all with the aid of a singular pre-trained model. Our findings affirm that the diversity inherent in our emotional labels leads to an enhancement in performance compared to previous distant supervision methods.

1 Introduction

This paper addresses the challenge that numerous Natural Language Processing (NLP) tasks face due to the lack of sufficient manually annotated data. Consequently, emotional expressions that co-occur with text have been utilized for distant supervision in sentiment analysis and related tasks within social media. This allows models to acquire valuable text representations before directly modeling these specific tasks. For example, state-of-the-art methods for sentiment analysis in social media frequently use positive and negative emoticons to train their models. Similarly, in prior research, hashtags like #anger, #joy, #happytweet, #ugh, #yuck, and #fml have been categorized into emotional labels for use in emotion analysis.

The practice of using distant supervision on noisy labels often leads to enhanced performance in the target task. In this paper, we present evidence that expanding distant supervision to a more varied selection of noisy labels enables models to develop more detailed representations of emotional content in text. This, in turn, improves performance on benchmark datasets designed for the detection of sentiment, emotions, and sarcasm. We further demonstrate that the representations learned by a single pre-trained model can be successfully generalized across five different domains.

Table 1 showcases example sentences which were scored by our model. For every sentence, the five most probable emojis are displayed, alongside the model's estimated probabilities.

Emojis do not always function as straightforward labels of emotional content. For instance, a positive emoji might clarify an ambiguous sentence or supplement text that might otherwise be seen as somewhat negative. While this is true, our results demonstrate that emojis can still be used to accurately categorize the emotional content of texts in numerous scenarios. Our DeepMoji model, for instance, is able to capture various interpretations of the word 'love' and slang terms like 'this is the shit' as having positive connotations (as illustrated in Table 1). To enable others to explore the prediction capabilities of our model, we have made an online demonstration available at deepmoji.mit.edu.

Our work makes the following contributions: We demonstrate that a vast number of readily accessible emoji occurrences on Twitter can be used to pre-train models for richer emotional representation than is typically achieved through distant supervision. We then transfer this learned knowledge to target tasks using a novel layer-wise fine-tuning approach. This technique yields significant improvements over state-of-the-art methods in areas such as emotion, sarcasm, and sentiment detection. Through extensive analyses on the influence of pre-training, our results highlight that the variety present in our emoji set plays a crucial role in the transfer learning capabilities of our model. We have made our pre-trained DeepMoji model publicly available to aid in a range of NLP tasks.

2 Related work

The use of emotional expressions as noisy labels in text to address the scarcity of labels is not a new concept. Initially, binarized emoticons served as noisy labels, but subsequent research has utilized hashtags and emojis. Previous studies have always manually determined which emotional category each emotional expression should belong to. Prior efforts have made use of emotion theories, such as Ekman's six basic emotions and Plutchik's eight basic emotions.

Such manual categorization necessitates an understanding of the emotional content inherent to each expression, which can be challenging and time-consuming for complex emotional combinations. Furthermore, any manual selection and categorization carries the potential for misinterpretations and might overlook essential details concerning usage. In contrast, our methodology requires no prior knowledge of the corpus and can capture the diverse usage of 64 emoji types (Table 1 presents examples, and Figure 3 shows how the model implicitly organizes emojis).

An alternative approach to automatically interpreting the emotional content of an emoji involves learning emoji embeddings from the words defining emoji-semantics, as found in official emoji tables. In our study, this approach has two significant limitations: (a) It requires emojis to be present during testing, whereas several domains have limited or no emoji usage. (b) The tables fail to capture the dynamic nature of emoji use, such as shifts in an emoji's intended meaning over time.

Knowledge from the emoji dataset can be transferred to target tasks in several ways. Multi-task learning, which involves training on multiple datasets at once, has been shown to have promising results. However, multi-task learning requires access to the emoji dataset whenever the classifier needs to be adjusted for a new target task. Requiring access to the dataset can be problematic when considering data access regulations. Data storage issues also arise, as the dataset used in this study comprises hundreds of millions of tweets (see Table 2). Instead, we use transfer learning which does not require access to the original dataset.

3 Method

3.1 Pretraining

In many instances, emojis function as a stand-in for the emotional content of text. Therefore, pre-training a model to predict which emojis were initially part of a text can improve performance in the target task. Social media contains many short texts that use emojis which can be used as noisy labels for pretraining. We used data from Twitter spanning from January 1, 2013, to June 1, 2017, but any data set containing emoji occurrences could be used.

The pretraining data set uses only English tweets that do not contain URLs. We think the content obtained from the URL is important for understanding the emotional content of the text in the tweet. Because of this we expect emojis associated with tweets containing URLs to be noisier labels than those in tweets without URLs, therefore the tweets with URLs have been removed.

Proper tokenization is crucial for generalization. All tweets are tokenized word-by-word. Words containing two or more repeated characters are shortened to the same token (for example, 'loool' and 'looooool' are tokenized as the same). We also use a special token for all URLs (which is relevant only for the benchmark datasets), user mentions (for example, '@acl2017' and '@emnlp2017' are treated the same), and numbers. To be included in the training set, a tweet must have at least one token that is not a punctuation mark, emoji, or special token.

Many tweets repeat the same emoji or contain multiple distinct emojis. To address this in our training data, for each unique emoji type, we save a separate tweet for pretraining, using that emoji type as the label. Regardless of the number of emojis associated with the tweet, we save only a single tweet for the pretraining for each unique emoji type. This pre-processing of data enables the pretraining to capture that multiple kinds of emotional content can be associated with the tweet. It also makes our pretraining task a single-label classification instead of a more complex multi-label classification.

To ensure that the pretraining encourages the models to learn a thorough understanding of the emotional content of text instead of just the emotional content associated with frequently used emojis, we create a balanced pretraining dataset. The pretraining data is split into training, validation, and test sets. The validation and test sets are randomly sampled such that each emoji is represented equally. The remaining data is upsampled to generate a balanced training dataset.

3.2 Model

With the availability of millions of emoji occurrences, we are able to train expressive classifiers with a limited risk of overfitting. We utilize a variant of the Long Short-Term Memory (LSTM) model, which has been successful in numerous NLP tasks. Our DeepMoji model uses an embedding layer with 256 dimensions to project each word into a vector space. A hyperbolic tangent activation function is used to ensure each embedding dimension remains within the range [-1, 1]. To understand each word in the context of the text, we use two bidirectional LSTM layers with 1024 hidden units each (512 in each direction). Lastly, we employ an attention layer that accepts all these layers as input through skip connections. (Figure 1 presents an illustration).

The attention mechanism enables the model to determine the importance of each word for the prediction task by weighting the words as it creates the text representation. A word like "amazing" is highly informative of the emotional meaning of a text and so should be treated accordingly. We use a basic method, taking inspiration from prior work, with a single parameter for each input channel:

$$e_i = h_i w_a \quad a_i = \frac{exp(e_i)}{\sum_{j=1} exp(e_j)} \quad v = \sum a_i h_i \tag{1}$$

Here, h_t stands for the representation of the word at time step t, and w_a is the weight matrix for the attention layer. The attention importance scores for each time step, a_t, are determined by multiplying the representations by the weight matrix, and then normalizing them to establish a probability distribution across the words. Finally, the text's representation vector, v, is found using a weighted summation over all time steps, with the attention importance scores used as weights. The representation vector that comes from the attention layer is a high-level encoding of the whole text. This is used as input into the final Softmax layer for classification. We have found that the addition of the attention mechanism and skip connections enhances the model's capabilities for transfer learning.

The only form of regularization used for the pretraining is L2 regularization with a coefficient of 10^{-6} on the embedding weights. For fine-tuning, further regularization is applied. We implemented our model using Theano and have made an easy-to-use version available that utilizes Keras.

3.3 Transfer learning

Our pre-trained model can be fine-tuned for a target task in several ways. Some methods involve 'freezing' layers by disabling parameter updates to prevent overfitting. One popular approach is to utilize the network as a feature extractor, where all model layers except the final one are frozen during fine-tuning (we will call this the "last" approach). An alternative method is to use the pre-trained model for initialization, where the full model is unfrozen (which we will refer to as the 'full' approach).

We put forward a new, simple transfer learning approach we are calling "chain-thaw." This approach sequentially unfreezes and fine-tunes one layer at a time. It increases accuracy on the target task, but requires more computational power for the fine-tuning process. By separately training each layer, the model can adjust individual patterns across the network while reducing the risk of overfitting. It appears that this sequential fine-tuning has a regularizing effect, similar to the layer-wise training explored for unsupervised learning.

More specifically, the chain-thaw approach starts by fine-tuning any new layers (often only a Softmax layer) to the target task until the validation set converges. Then, the approach individually fine-tunes each layer, starting with the first layer in the network. Lastly, the entire model is trained with all layers. Each time the model converges (as measured on the validation set), the weights are restored to their optimal setting, preventing overfitting in a similar manner to early stopping. Figure 2 illustrates this process. If only step a) in the figure is performed, this is the same as the 'last' approach, where the existing network is used as a feature extractor. Likewise, only performing step d) is the same as the 'full' approach, where the pre-trained weights are used as the initialization for a fully trainable network. While the chain-thaw procedure may seem extensive, it can be implemented with just a few lines of code. Also, the added time spent on fine-tuning is not large, when considering the use of GPUs on small datasets of manually annotated data which is often the case.

The chain-thaw approach has the benefit of expanding the vocabulary to new domains with a low risk of overfitting. For a given dataset, up to 10,000 new words from the training set are added to the vocabulary.

Table 2 shows the number of tweets in the pretraining dataset associated with each emoji in millions.

4 Experiments

4.1 Emoji prediction

We use a raw dataset of 56.6 billion tweets, which is filtered down to 1.2 billion relevant tweets. In the pretraining dataset, a single copy of a tweet is stored for every unique emoji, resulting in a dataset with 1.6 billion tweets. Table 2 shows the distribution of tweets across different emoji types. We used a validation set and a test set, both containing 640K tweets (10K of each emoji type), to evaluate performance on the pretraining task. The remaining tweets were used for the training set, which was balanced using upsampling.

The performance of the DeepMoji model on the pretraining task was evaluated, with the results shown in Table 3. We use both top 1 and top 5 accuracy for the evaluation as the emoji labels are noisy and multiple emojis can potentially be appropriate for a given sentence. For comparison purposes, we also train a version of our DeepMoji model with smaller LSTM layers and a bag-of-words classifier, fastText, which has recently shown competitive results. We use a 256 dimension vector for the fastText classifier, making it almost identical to only using the embedding layer from the DeepMoji model. The difference in top 5 accuracy between the fastText classifier (36.2%) and the largest DeepMoji model (43.8%) highlights the difficulty of the emoji prediction task. Since the two classifiers only differ in that the DeepMoji model has LSTM layers and an attention layer between the embedding and the Softmax layer, this difference in accuracy demonstrates the importance of capturing each word's context.

Table 3 displays the accuracy of classifiers on the emoji prediction task. The value d refers to the dimensionality of each LSTM layer and the parameters are given in millions.

Model	Params	Top 1	Top 5
Random	-	1.6%	7.8%
fasttext	12.8	12.8%	36.2%
DeepMoji (d=512)	15.5	16.7%	43.3%
DeepMoji (d=1024)	22.4	17.0%	43.8%

Table 1: Accuracy of classifiers on the emoji prediction task. d refers to the dimensionality of each LSTM layer. Parameters are in millions.

4.2 Benchmarking

We evaluate our method on 3 distinct NLP tasks using 8 datasets across 5 domains. For fair comparison, DeepMoji is compared to other methods that utilize external data sources in addition to the benchmark dataset. We used an averaged F1 measure across classes for evaluating emotion analysis and sarcasm detection, as these consist of unbalanced datasets. Sentiment datasets are evaluated using accuracy.

Many benchmark datasets have an issue with data scarcity, especially in emotion analysis. Many studies that introduce new methods for emotion analysis often evaluate their performance on a single benchmark dataset, SemEval 2007 Task 14, which contains only 1250 data points. There has been criticism regarding the use of correlation with continuous ratings as a measure, making only the somewhat limited binary evaluation possible. We only evaluate the emotions Fear, Joy, Sadness because the remaining emotions are found in less than 5

To fully assess our method on emotion analysis, we make use of two other datasets. First, a dataset of emotions in tweets about the Olympic Games, created by Sintsova et al. which we convert to a single-label classification task. Second, a dataset of self-reported emotional experiences from a large group of psychologists. Because these two datasets have not been evaluated in prior work, we compare against a state-of-the-art approach based on a valence-arousal-dominance framework. The scores extracted using this framework are mapped to the classes in the datasets using logistic regression with cross-validation parameter optimization. We have made our preprocessing code available so that these two datasets may be used for future benchmarking in emotion analysis.

We assessed the performance of sentiment analysis using three benchmark datasets. These small datasets were chosen to highlight the significance of the transfer learning capabilities of the evaluated models. Two datasets, SS-Twitter and SS-Youtube, are from SentiStrength and follow the relabeling as described by prior work to create binary labels. The third dataset is from SemEval 2016 Task4A. Because tweets are often deleted from Twitter, the SemEval dataset has experienced data decay. This makes comparisons across papers difficult. Approximately 15

The current state of the art in sentiment analysis on social media (and winner of SemEval 2016 Task 4A) uses an ensemble of convolutional neural networks that are pre-trained on a private dataset of tweets with emoticons. This makes it difficult to replicate. As a substitute, we pre-train a model that uses the hyperparameters of the largest model in their ensemble on the positive/negative emoticon dataset. Using this pretraining as an initialization, we fine-tune the model on the target tasks, utilizing early stopping based on a validation set. We implemented Sentiment-Specific Word Embeddings (SSWE), using embeddings available on the authors' website, but found that it performed worse than the pretrained convolutional neural network, and these results have been excluded.

Table 4 presents a description of the benchmark datasets. Datasets that did not have pre-existing training/test splits were split by us, and these splits are publicly available. Data from the training set was used for hyperparameter tuning.

Identifier	Study	Task	Domain	Classes	Ntrain	Ntest
SE0714	(Strapparava and Mihalcea, 2007)	Emotion	Headlines	3	250	1000
Olympic	(Sintsova et al., 2013)	Emotion	Tweets	4	250	709
PsychExp	(Wallbott and Scherer, 1986)	Emotion	Experiences	7	1000	6480
SS-Twitter	(Thelwall et al., 2012)	Sentiment	Tweets	2	1000	1113
SS-Youtube	(Thelwall et al., 2012)	Sentiment	Video Comments	2	1000	1142
SE1604	(Nakov et al., 2016)	Sentiment	Tweets	3	7155	31986
SCv1	(Walker et al., 2012)	Sarcasm	Debate Forums	2	1000	995
SCv2-GEN	(Oraby et al., 2016)	Sarcasm	Debate Forums	2	1000	2260

Table 2: Description of benchmark datasets. Datasets without pre-existing training/test splits are split by us (with splits publicly available). Data used for hyperparameter tuning is taken from the training set.

For sarcasm detection, we used versions 1 and 2 of the sarcasm dataset from the Internet Argument Corpus. It should be noted that the results from these benchmarks that are shown elsewhere are not directly comparable, as only a subset of the data is available online. We establish a state-of-the-art baseline by modeling embedding-based features alongside unigrams, bigrams, and trigrams with an SVM. GoogleNews word2vec embeddings are used to compute the embedding-based features. Cross-validation was used to perform a hyperparameter search for regularization parameters. The sarcasm dataset version 2 includes both a quoted text and a sarcastic response, but only the response was used to keep models consistent across the datasets.

Table 5 displays a comparison across benchmark datasets. The reported values are averages across 5 runs. Variations refer to the transfer learning approaches that we discussed, and 'new' refers to a model trained without pretraining.

Dataset	Measure	State of the art	DeepMoji (new)	DeepMoji (full)	DeepMoji (last)	DeepMoji (chain-thaw)
SE0714	F1	.34	.21	.31	.36	.37
Olympic	F1	.50	.43	.50	.61	.61
PsychExp	F1	.45	.32	.42	.56	.57
SS-Twitter	Acc	.82	.62	.85	.87	.88
SS-Youtube	Acc	.86	.75	.88	.92	.93
SE1604	Acc	.51	.51	.54	.58	.58
SCv1	F1	.63	.67	.65	.68	.69
SCv2-GEN	F1	.72	.71	.71	.74	.75

Table 3: Comparison across benchmark datasets. Reported values are averages across five runs. Variations refer to transfer learning approaches with 'new' being a model trained without pretraining.

We used the Adam optimizer for training, with the gradient norm clipped to 1. For training all new layers, we set the learning rate to 10^{-3} and to 10^{-4} when fine-tuning any pre-trained layers. To prevent overfitting on the small datasets, 10

Table 5 demonstrates that the DeepMoji model outperforms the state of the art across all the benchmark datasets and that our new 'chain-thaw' method yields the highest transfer learning performance. The results are averaged across 5 runs to reduce the variance. We confirm statistical significance using bootstrap testing with 10,000 samples, our model performance was statistically better than the state-of-the-art across all benchmark datasets ($p < 0.001$).

Our model exceeds the performance of the state of the art even on datasets that come from different domains than the tweets that the model was pre-trained on. A crucial difference between the pretraining dataset and the benchmark datasets is the length of the observations. The average number of tokens per tweet in the pretraining dataset is 11. Meanwhile, board posts from the Internet Argument Corpus version 1 (for example), have an average of 66 tokens, with some posts being much longer.

5 Model Analysis

5.1 Importance of emoji diversity

A key difference between this work and prior research that used distant supervision is the variety in noisy labels. For example, other studies only used positive and negative emoticons as noisy labels. Other studies used more nuanced sets of noisy labels, but our set is the most varied known to us. To investigate the effect of using a diverse set of emojis, we created a subset of our pretraining data that included tweets with one of 8 emojis, which are similar to the positive/negative emoticons used in other work. Because the dataset based on this reduced set of emojis contains 433 million tweets, any performance differences on benchmark datasets are more likely linked to the diversity of the labels than to differences in dataset sizes.

We trained our DeepMoji model to predict whether tweets contained positive or negative emojis, and we evaluated this pre-trained model on benchmark datasets. We call this the DeepMoji-PosNeg model. To assess the emotional representations learned by the two pre-trained models, we used the 'last' transfer learning approach to allow the models to map already learned features to classes in the

target datasets. Table 6 shows that DeepMoji-PosNeg performs worse than DeepMoji across all 8 benchmarks. This demonstrates that the diversity of our emoji types enables the model to acquire richer representations of emotional content in text, which in turn is more useful for transfer learning.

Table 6 compares benchmarks using a smaller emoji set (Pos/Neg emojis) or a standard architecture (standard LSTM). Results for DeepMoji from Table 5 have been added for comparison. The evaluation metrics are the same as in Table 5. Reported values are averages across 5 runs.

Dataset	Pos/Neg emojis	Standard LSTM	DeepMoji
SE0714	.32	.35	.36
Olympic	.55	.57	.61
PsychExp	.40	.49	.56
SS-Twitter	.86	.86	.87
SS-Youtube	.90	.91	.92
SE1604	.56	.57	.58
SCv1	.66	.66	.68
SCv2-GEN	.72	.73	.74

Table 4: Benchmarks using a smaller emoji set (Pos/Neg emojis) or a classic architecture (standard LSTM). Results for DeepMoji from Table 5 are added for convenience. Evaluation metrics are as in Table 5. Reported values are the averages across five runs.

Many emojis express similar emotional content, but have subtle variations in usage that our model can capture. By using hierarchical clustering on the correlation matrix of the DeepMoji model's predictions on the test set, we can see that the model captures many expected similarities (Figure 3). For example, the model groups emojis into broad categories related to negativity, positivity, or love. It also differentiates within these categories. For example, mapping sad emojis to one subcategory of negativity, annoyed emojis to another subcategory, and angry emojis to a third.

5.2 Model architecture

Our DeepMoji model architecture employs an attention mechanism and skip connections, which assist in transferring learned representations to new domains and tasks. Here, we compare the DeepMoji model architecture to a standard 2-layer LSTM. Both were compared using the 'last' transfer learning approach, and all regularization and training parameters were consistent.

Table 6 shows that the DeepMoji model performs better than a standard 2-layer LSTM across all the benchmark datasets. These two architectures performed equally on the pretraining task. This indicates that the DeepMoji model architecture is better for transfer learning, even if it is not necessarily better for a single supervised classification task with an abundance of available data.

We believe that the improvements in transfer learning can be attributed to two factors: (a) The attention mechanism with skip connections provides straightforward access to learned low-level features for any time step, making it easy to use this information if needed for a new task. (b) The skip connections improve the gradient flow from the output layer to the early layers in the network. This is useful when parameters in early layers are adjusted as a part of transfer learning to small datasets. Further analysis of these factors in future work would allow us to confirm why our architecture outperforms a standard 2-layer LSTM.

5.3 Analyzing the effect of pretraining

The target task's performance benefits significantly from pretraining, as shown in Table 5. Here, we separate the effects of pretraining into two factors: word coverage and phrase coverage. These two effects provide regularization to the model, preventing overfitting (the supplementary material includes a visualization of this regularization).

There are multiple ways of expressing sentiment, emotion, or sarcasm. Because of this, the test set may contain language use not present in the training set. Pretraining helps the target task models focus on low-support evidence by having already seen similar language in the pretraining dataset. To examine this effect, we measure the improvement in word coverage on the test set when using

pretraining. Word coverage is defined as the percentage of words in the test dataset that were also seen in the training/pretraining dataset (as shown in Table 7). One key reason that the 'chain-thaw' approach outperforms other transfer learning approaches is its ability to tune the embedding layer with a low risk of overfitting. Table 7 shows how adding new words to the vocabulary as part of the tuning process increased word coverage.

It is important to note that word coverage can be misleading in this context. In many small datasets, a word may occur only once in the training set. In contrast, all the words in the pretraining vocabulary are present in thousands or even millions of observations, enabling the model to learn a good representation of the emotional and semantic meaning. Therefore, the benefits of pretraining for word representations likely extend beyond the differences seen in Table 7.

Table 7 shows the word coverage on benchmark test sets. This compares the use of only the vocabulary generated by finding words in the training data ('own'), the pretraining vocabulary ('last'), or a combination of both vocabularies ('full / chain-thaw').

Dataset	Own	Last	Full / Chain-thaw
SE0714	41.9%	93.6%	94.0%
Olympic	73.9%	90.3%	96.0%
PsychExp	85.4%	98.5%	98.8%
SS-Twitter	80.1%	97.1%	97.2%
SS-Youtube	79.6%	97.2%	97.3%
SE1604	86.1%	96.6%	97.0%
SCv1	88.7%	97.3%	98.0%
SCv2-GEN	86.5%	97.2%	98.0%

Table 5: Word coverage on benchmark test sets using only the vocabulary generated by finding words in the training data ('own'), the pretraining vocabulary ('last') or a combination of both vocabularies ('full / chain-thaw').

To analyze how important capturing phrases and the context of each word are, we evaluated the accuracy on the SS-Youtube dataset using a fastText classifier that was pre-trained using the same emoji dataset as our DeepMoji model. This fastText classifier is similar to only using the embedding layer from the DeepMoji model. We then evaluated the representations learned by fine-tuning the models as feature extractors (using the 'last' transfer learning approach). The fastText model achieved an accuracy of 63

5.4 Comparing with human-level agreement

To see how well our DeepMoji classifier performs compared to humans, we created a dataset of randomly selected tweets that were annotated for sentiment. Each tweet was annotated by a minimum of 10 English-speaking Amazon Mechanical Turkers (MTurks) who lived in the USA. The tweets were rated on a scale from 1 to 9, with a 'Do not know' option. Guidelines were provided to the human raters. The tweets were selected to contain only English text and no mentions or URLs, so they could be rated without extra contextual information. Tweets where more than half the evaluators chose 'Do not know' were removed (98 tweets).

For every tweet, we randomly select a single MTurk rating as the 'human evaluation.' We average the remaining nine MTurk ratings to make the ground truth. The 'sentiment label' for a given tweet is thus defined as the overall consensus among raters, excluding the randomly selected 'human evaluation' rating. To ensure clear separation between the label categories, we removed neutral tweets that fell within the interval [4.5, 5.5] (roughly 29

Table 8 shows that the agreement of the random MTurk rater is 76.1

Table 8 compares the agreement between classifiers and the aggregate opinion of Amazon Mechanical Turkers on sentiment prediction of tweets.

Model	Agreement
Random	50.1%
fastText	71.0%
MTurk	76.1%
DeepMoji	82.4%

Table 6: Comparison of agreement between classifiers and the aggregate opinion of Amazon Mechanical Turkers on sentiment prediction of tweets.

6 Conclusion

We have demonstrated how the abundance of text on social media containing emojis can be used to pre-train models. This enables them to acquire representations of emotional content in text. Our findings demonstrate that the diversity of our emoji set is crucial to our method's performance. This was found by comparing the model performance against an identical model that was pre-trained on a subset of emojis. Our pre-trained DeepMoji model is available for other researchers to use for diverse emotion-related NLP tasks.