

Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm

Bjarke Felbo¹, Alan Mislove², Anders Søgaard³, Iyad Rahwan¹, Sune Lehmann⁴

¹Media Lab, Massachusetts Institute of Technology

²College of Computer and Information Science, Northeastern University

³Department of Computer Science, University of Copenhagen

⁴DTU Compute, Technical University of Denmark

Abstract

NLP tasks are often limited by scarcity of manually annotated data. In social media sentiment analysis and related tasks, researchers have therefore used binarized emoticons and specific hashtags as forms of distant supervision. Our paper shows that by extending the distant supervision to a more diverse set of noisy labels, the models can learn richer representations. Through emoji prediction on a dataset of 1246 million tweets containing one of 64 common emojis we obtain state-of-the-art performance on 8 benchmark datasets within emotion, sentiment and sarcasm detection using a single pretrained model. Our analyses confirm that the diversity of our emotional labels yield a performance improvement over previous distant supervision approaches.




































1 Introduction

A variety of NLP tasks are limited by scarcity of manually annotated data. Therefore, co-occurring emotional expressions have been used for distant supervision in social media sentiment analysis and related tasks to make the models learn useful text representations before modeling these tasks directly. For instance, the state-of-the-art approaches within sentiment analysis of social media data use positive/negative emoticons for training their models (Deriu et al., 2016; Tang et al., 2014). Similarly, hashtags such as #anger, #joy, #happytweet, #ugh, #yuck and #fml have in previous research been mapped into emotional categories for emotion analysis (Mohammad, 2012).

Distant supervision on noisy labels often enables a model to obtain better performance on the target task. In this paper, we show that extend-

ing the distant supervision to a more diverse set of noisy labels enables the models to learn richer representations of emotional content in text, thereby obtaining better performance on benchmarks for detecting sentiment, emotions and sarcasm. We show that the learned representation of a single pretrained model generalizes across 5 domains.

Table 1: Example sentences scored by our model. For each text the top five most likely emojis are shown with the model’s probability estimates.

| | | | | | |
|---------------------------------------|---|---|---|---|---|
| I love mom’s cooking |  |  |  |  |  |
| | 49.1% | 8.8% | 3.1% | 3.0% | 2.9% |
| I love how you never reply back.. |  |  |  |  |  |
| | 14.0% | 8.3% | 6.3% | 5.4% | 5.1% |
| I love cruising with my homies |  |  |  |  |  |
| | 34.0% | 6.6% | 5.7% | 4.1% | 3.8% |
| I love messing with yo mind!! |  |  |  |  |  |
| | 17.2% | 11.8% | 8.0% | 6.4% | 5.3% |
| I love you and now you’re just gone.. |  |  |  |  |  |
| | 39.1% | 11.0% | 7.3% | 5.3% | 4.5% |
| This is shit |  |  |  |  |  |
| | 7.0% | 6.4% | 6.0% | 6.0% | 5.8% |
| This is the shit |  |  |  |  |  |
| | 10.9% | 9.7% | 6.5% | 5.7% | 4.8% |

Emojis are not always a direct labeling of emotional content. For instance, a positive emoji may serve to disambiguate an ambiguous sentence or to complement an otherwise relatively negative text. Kunneman et al. (2014) discuss a similar duality in the use of emotional hashtags such as #nice and #lame. Nevertheless, our work shows that emojis can be used to classify the emotional content of texts accurately in many cases. For instance, our DeepMoji model captures varied usages of the word ‘love’ as well as slang such as ‘this is the shit’ being a positive statement (see Table 1). We provide an online demo at deepmoji.mit.edu to allow others to explore the predictions of our model.

Contributions We show how millions of readily available emoji occurrences on Twitter can be used to pretrain models to learn a richer emotional

representation than traditionally obtained through distant supervision. We transfer this knowledge to the target tasks **using a new layer-wise fine-tuning** method, obtaining improvements over the state-of-the-art within a range of tasks: emotion, sarcasm and sentiment detection. We present multiple analyses on the effect of pretraining, including results that show that the diversity of our emoji set is important for the transfer learning potential of our model. Our pretrained DeepMojji model is released with the hope that other researchers can use it for various NLP tasks¹.

2 Related work

Using emotional expressions as noisy labels in text to counter scarcity of labels is not a new idea (Read, 2005; Go et al., 2009). Originally, binarized emoticons were used as noisy labels, but later also hashtags and emojis have been used. To our knowledge, previous research has always manually specified which emotional category each emotional expression belong to. Prior work has used theories of emotion such as Ekman’s six basic emotions and Plutchik’s eight basic emotions (Mohammad, 2012; Suttles and Ide, 2013).

Such manual categorization requires an understanding of the emotional content of each expression, which is difficult and time-consuming for sophisticated combinations of emotional content. Moreover, any manual selection and categorization is prone to misinterpretations and may omit important details regarding usage. In contrast, our approach requires no prior knowledge of the corpus and can capture diverse usage of 64 types of emojis (see Table 1 for examples and Figure 3 for how the model implicitly groups emojis).

Another way of automatically interpreting the emotional content of an emoji is to learn emoji embeddings from the words describing the emoji-semantics in official emoji tables (Eisner et al., 2016). This approach, in our context, suffers from two severe limitations: a) It requires emojis at test time while there are many domains with limited or no usage of emojis. b) The tables do not capture the dynamics of emoji usage, i.e., drift in an emoji’s intended meaning over time.

Knowledge can be transferred from the emoji dataset to the target task in many different ways. In particular, multitask learning with simultaneous

¹Available with preprocessing code, examples of usage, benchmark datasets etc. at github.com/bfelbo/DeepMojji

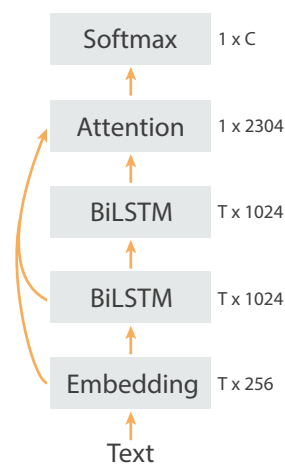


Figure 1: Illustration of the DeepMojji model with S being text length and C the number of classes.

training on multiple datasets has shown promising results (Collobert and Weston, 2008). However, multitask learning requires access to the emoji dataset whenever the classifier needs to be tuned for a new target task. Requiring access to the dataset is problematic in terms of violating data access regulations. There are also issues from a data storage perspective as the dataset used for this research contains hundreds of millions of tweets (see Table 2). Instead we use transfer learning (Bengio et al., 2012) as described in §3.3, which does not require access to the original dataset, but only the pretrained classifier.

3 Method

3.1 Pretraining

In many cases, emojis serve as a proxy for the emotional contents of a text. Therefore, pretraining on the classification task of predicting which emoji were initially part of a text can improve performance on the target task (see §5.3 for an analysis of why our pretraining helps). Social media contains **large amounts of short texts with emojis** that can be utilized as **noisy** labels for pretraining. Here, we use data from Twitter from **January 1st 2013 to June 1st 2017**, but any dataset with emoji occurrences could be used.

Only English tweets without URL’s are used for the pretraining dataset. Our hypothesis is that the content obtained from the URL is likely to be important for understanding the emotional content of the text in the tweet. Therefore, we expect emojis associated with these tweets to be noiser labels

than for tweets without URLs, and **the tweets with URLs are thus removed**.

Proper tokenization is important for generalization. **All tweets are tokenized on a word-by-word basis**. Words with 2 or more repeated characters are shortened to the **same token** (e.g. 'loool' and 'loooooool' are tokenized such that they are treated the same). Similarly, we use a special token for all URLs (only relevant for benchmark datasets), user mentions (e.g. '@acl2017' and '@emnlp2017' are **thus treated the same**) and numbers. To be included in the training set the tweet must contain at least 1 token that is not a punctuation symbol, emoji or special token².

Many tweets contain multiple repetitions of the same emoji or multiple different emojis. In the training data, we address this in the following way. For each unique emoji type, we save a separate tweet for the pretraining with that emoji type as the label. We only save a single tweet for the pretraining per unique emoji type regardless of the number of emojis associated with the tweet. This data pre-processing allows the pretraining task to capture that multiple types of emotional content are associated with the tweet while making our pretraining task a single-label classification instead of a more complicated multi-label classification.

To ensure that the pretraining encourages the models to learn a rich understanding of emotional content in text rather than only emotional content associated with the most used emojis, we create a balanced pretraining dataset. The pretraining data is split into a training, validation and test set, where the validation and test set is randomly sampled in such a way that each emoji is equally represented. The remaining data is upsampled to create a balanced training dataset.

3.2 Model

With the millions of emoji occurrences available, we can train very expressive classifiers with limited risk of overfitting. We use a variant of the **Long Short-Term Memory (LSTM) model that has been successful at many NLP tasks** (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014). Our DeepMoji model uses an embedding layer of 256 dimensions to project each word into a vector space. A hyperbolic **tangent activation function** is used to enforce a constraint of each embedding dimension being within **(-1, 1)**. To capture the con-

text of each word we use **two bidirectional LSTM** layers with 1024 hidden units in each (512 in each direction). Finally, an **attention layer** that take all of these layers as input using **skip-connections** is used (see Figure 1 for an illustration).

The attention mechanism lets the model decide the importance of each word for the prediction task by weighing them when constructing the representation of the text. For instance, a word such as 'amazing' is likely to be very informative of the emotional meaning of a text and it should thus be treated accordingly. We use a simple approach inspired by (Bahdanau et al., 2014; Yang et al., 2016) with a single parameter pr. input channel:

$$\begin{aligned} e_t &= h_t w_a \\ a_t &= \frac{\exp(e_t)}{\sum_{i=1}^T \exp(e_i)} \\ v &= \sum_{i=1}^T a_i h_i \end{aligned}$$

Here h_t is 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 obtained by multiplying the representations with the weight matrix and then normalizing to construct a probability distribution over the words. **Lastly, the representation vector for the text, v , is found by a weighted summation over all the time steps using the attention importance scores as weights**. This representation vector obtained from the attention layer is a **high-level encoding of the entire text**, which is used as input to the final **Softmax** layer for **classification**. We find that adding the attention mechanism and skip-connections improves the model's capabilities for transfer learning (see §5.2 for more details).

The only regularization used for the pretraining task is a **L2** regularization of $1E-6$ on the embedding weights. For the finetuning additional regularization is applied (see §4.2). Our model is implemented using **Theano** (Theano Development Team, 2016) and we make an easy-to-use version available that uses **Keras** (Chollet et al., 2015).

3.3 Transfer learning

Our pretrained model can be **fine-tuned** to the target task in multiple ways with some approaches 'freezing' layers by disabling parameters updates to prevent overfitting. One common approach is

²Details available at github.com/bfelbo/deepmoji

to use the network as a feature extractor (Donahue et al., 2014), where all layers in the model are frozen when fine-tuning on the target task except the last layer (hereafter referred to as the ‘last’ approach). Alternatively, another common approach is to use the pretrained model as an initialization (Erhan et al., 2010), where the full model is unfrozen (hereafter referred to as ‘full’).

We propose a new simple transfer learning approach, ‘chain-thaw’, that sequentially unfreezes and fine-tunes a single layer at a time. This approach increases accuracy on the target task at the expense of extra computational power needed for the fine-tuning. By training each layer separately the model is able to adjust the individual patterns across the network with a reduced risk of overfitting. The sequential fine-tuning seems to have a regularizing effect similar to what has been examined with layer-wise training in the context of unsupervised learning (Erhan et al., 2010).

More specifically, the chain-thaw approach first fine-tunes any new layers (often only a Softmax layer) to the target task until convergence on a validation set. Then the approach fine-tunes each layer individually starting from 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 reloaded to the best setting, thereby preventing overfitting in a similar manner to early stopping (Sjöberg and Ljung, 1995). This process is illustrated in Figure 2. Note how only performing step a) in the figure is identical to the ‘last’ approach, where the existing network is used as a feature extractor. Similarly, only doing step d) is identical to the ‘full’ approach, where the pretrained weights are used as an initialization for a fully trainable network. Although the chain-thaw procedure may seem extensive it is easily implemented with only a few lines of code. Similarly, the additional time spent on fine-tuning is limited when the target task uses GPUs on small datasets of manually annotated data as is often the case.

A benefit of the chain-thaw approach is the ability to expand the vocabulary to new domains with little risk of overfitting. For a given dataset up to 10000 new words from the training set are added to the vocabulary. §5.3 contains analysis on the added word coverage gained from this approach.

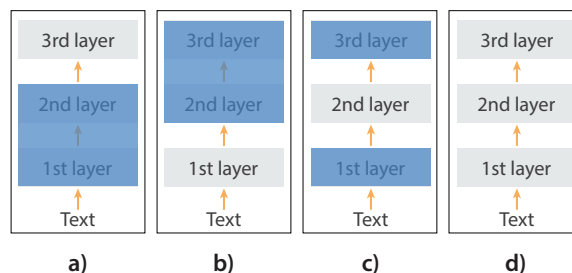


Figure 2: Illustration of the chain-thaw transfer learning approach, where each layer is fine-tuned separately. Layers covered with a blue rectangle are frozen. Step a) tunes any new layers, b) then tunes the 1st layer and c) the next layer until all layers have been fine-tuned individually. Lastly, in step d) all layers are fine-tuned together.

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

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|-------|----|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|---|------|
| 🤔 | 233.7 | ❤️ | 82.2 | 👉 | 79.5 | 👍 | 78.1 | 👎 | 60.8 | 😬 | 54.7 | 😏 | 54.6 | 😐 | 51.7 | 👏 | 50.5 | 👀 | 44.0 | 👉 | 39.5 | 👍 | 39.1 | 👎 | 34.8 | 😬 | 34.4 | 😐 | 32.1 | 👏 | 28.1 |
| 🙏 | 24.8 | 👉 | 23.4 | 👍 | 21.6 | 👎 | 21.0 | 👏 | 20.5 | 👀 | 20.3 | 👉 | 19.9 | 👍 | 19.6 | 👎 | 18.9 | 👏 | 17.5 | 👀 | 17.0 | 👉 | 16.9 | 👍 | 16.1 | 👎 | 15.3 | 👏 | 15.2 | 👀 | 15.0 |
| 👉 | 14.9 | 👍 | 14.3 | 👎 | 14.2 | 👏 | 14.2 | 👀 | 12.9 | 👉 | 12.4 | 👍 | 12.0 | 👎 | 12.0 | 👏 | 11.7 | 👀 | 11.7 | 👉 | 11.3 | 👍 | 11.2 | 👎 | 11.1 | 👏 | 11.0 | 👀 | 11.0 | 👉 | 10.8 |
| 👍 | 10.2 | 👎 | 9.6 | 👏 | 9.5 | 👀 | 9.3 | 👉 | 9.2 | 👍 | 8.9 | 👎 | 8.7 | 👏 | 8.6 | 👀 | 8.1 | 👉 | 6.3 | 👍 | 6.0 | 👎 | 5.7 | 👏 | 5.6 | 👀 | 5.5 | 👉 | 5.4 | 👍 | 5.1 |

4 Experiments

4.1 Emoji prediction

We use a raw dataset of 56.6 billion tweets, which is then filtered to 1.2 billion relevant tweets (see details in §3.1). In the pretraining dataset a copy of a single tweet is stored once for each unique emoji, resulting in a dataset consisting of 1.6 billion tweets. Table 2 shows the distribution of tweets across different emoji types. To evaluate performance on the pretraining task a validation set and a test set both containing 640K tweets (10K of each emoji type) are used. The remaining tweets are used for the training set, which is balanced using upsampling.

The performance of the DeepMoji model is evaluated on the pretraining task with the results shown in Table 3. Both top 1 and top 5 accuracy is used for the evaluation as the emoji labels are noisy with multiple emojis being potentially correct for any given sentence. For comparison we also train a version of our DeepMoji model with smaller LSTM layers and a bag-of-words classifier, fastText, that has recently shown competitive results (Joulin et al., 2016). We use 256 dimen-

Table 3: Accuracy of classifiers on the emoji prediction task. d refers to the dimensionality of each LSTM layer. Parameters are in millions.

| | 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% |

sions for this fastText classifier, thereby 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%) underlines the difficulty of the emoji prediction task. As the two classifiers only differ in that the DeepMoji model has LSTM layers and an attention layer between the embedding and Softmax layer, this difference in accuracy demonstrates the importance of capturing the context of each word.

4.2 Benchmarking

We benchmark our method on 3 different NLP tasks using 8 datasets across 5 domains. To make for a fair comparison, we compare DeepMoji to other methods that also utilize external data sources in addition to the benchmark dataset. An averaged F1-measure across classes is used for evaluation in emotion analysis and sarcasm detection as these consist of unbalanced datasets while sentiment datasets are evaluated using accuracy.

An issue with many of the benchmark datasets is data scarcity, which is particularly problematic within emotion analysis. Many recent papers proposing new methods for emotion analysis such as (Staiano and Guerini, 2014) only evaluate performance on a single benchmark dataset, SemEval 2007 Task 14, that contains 1250 observations. Recently, criticism has been raised concerning the use of correlation with continuous ratings as a measure (Buechel and Hahn, 2016), making only the somewhat limited binary evaluation possible. We only evaluate the emotions {Fear, Joy, Sadness} as the remaining emotions occur in less than 5% of the observations.

To fully evaluate our method on emotion analysis against the current methods we thus make use of two other datasets: A dataset of emotions in tweets related to the Olympic Games created by Sintsova et al. that we convert to a single-label

classification task and a dataset of self-reported emotional experiences created by a large group of psychologists (Wallbott and Scherer, 1986). See the supplementary material for details on the datasets and the preprocessing. As these two datasets do not have prior evaluations, we evaluate against a state-of-the-art approach, which is based on a valence-arousal-dominance framework (Buechel and Hahn, 2016). The scores extracted using this approach are mapped to the classes in the datasets using a logistic regression with parameter optimization using cross-validation. We release our preprocessing code and hope that these 2 two datasets will be used for future benchmarking within emotion analysis.

We evaluate sentiment analysis performance on three benchmark datasets. These small datasets are chosen to emphasize the importance of the transfer learning ability of the evaluated models. Two of the datasets are from SentiStrength (Thelwall et al., 2010), SS-Twitter and SS-Youtube, and follow the relabeling described in (Saif et al., 2013) to make the labels binary. The third dataset is from SemEval 2016 Task4A (Nakov et al., 2016). Due to tweets being deleted from Twitter, the SemEval dataset suffers from data decay, making it difficult to compare results across papers. At the time of writing, roughly 15% of the training dataset for SemEval 2016 Task 4A was impossible to obtain. We choose not to use review datasets for sentiment benchmarking as these datasets contain so many words pr. observation that even bag-of-words classifiers and unsupervised approaches can obtain a high accuracy (Joulin et al., 2016; Radford et al., 2017).

The current state of the art for sentiment analysis on social media (and winner of SemEval 2016 Task 4A) uses an ensemble of convolutional neural networks that are pretrained on a private dataset of tweets with emoticons, making it difficult to replicate (Deriu et al., 2016). Instead we pretrain a model with the hyperparameters of the largest model in their ensemble on the positive/negative emoticon dataset from Go et al. (2009). Using this pretraining as an initialization we finetune the model on the target tasks using early stopping on a validation set to determine the amount of training. We also implemented the Sentiment-Specific Word Embedding (SSWE) using the embeddings available on the authors’ website (Tang et al., 2014), but found that it performed worse

Table 4: 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.

| Identifier | Study | Task | Domain | Classes | N_{train} | N_{test} |
|------------|----------------------------------|-----------|----------------|---------|-------------|------------|
| 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 5: Comparison across benchmark datasets. Reported values are **averages** across **five runs**. Variations refer to transfer learning approaches in §3.3 with ‘new’ being a model trained without pretraining.

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

than the pretrained convolutional neural network. These results are therefore excluded.

For sarcasm detection we use the sarcasm dataset version 1 and 2 from the Internet Argument Corpus (Walker et al., 2012). Note that results presented on these benchmarks in e.g. Oraby et al. (2016) are not directly comparable as only a subset of the data is available online.³ A state-of-the-art baseline is found by modeling the embedding-based features from Joshi et al. (2016) alongside **unigrams**, **bigrams** and **trigrams** with an **SVM**. GoogleNews **word2vec** embeddings (Mikolov et al., 2013) are used for computing the embedding-based features. A hyperparameter search for regularization parameters is carried out using cross-validation. Note that the sarcasm dataset version 2 contains both a quoted text and a sarcastic response, but to keep the models identical across the datasets only the response is used.

For training we use the **Adam optimizer** (Kingma and Ba, 2015) with **gradient clipping** of the norm to **1**. Learning rate is set to **1E-3** for training of all new layers and **1E-4**

³We contacted the authors, but were unable to obtain the full dataset for neither version 1 or version 2.

for finetuning any pretrained layers. To prevent overfitting on the small datasets, 10% of the channels across all words in the embedding layer are dropped out during training. Unlike e.g. (Gal and Ghahramani, 2016) we do not drop out entire words in the input as some of our datasets contain observations with so few words that it could change the meaning of the text. In addition to the embedding dropout, L2 regularization for the embedding weights is used and 50% dropout is applied to the penultimate layer.

Table 5 shows that the DeepMoji model outperforms the state of the art across all benchmark datasets and that our new ‘chain-thaw’ approach consistently yields the highest performance for the transfer learning, albeit often only slightly better or equal to the ‘last’ approach. Results are averaged across 5 runs to reduce the variance. We test the statistical significance of our results by comparing the performance of DeepMoji (chain-thaw) vs. the state of the art. Bootstrap testing with 10000 samples is used. On all datasets are our results statistically significantly better than the state of the art with $p < 0.001$.

Our model is able to out-perform the state-of-

the-art on datasets that originate from domains that differ substantially from the tweets on which it was pretrained. A key difference between the pre-training dataset and the benchmark datasets is the length of the observations. The average number of tokens pr. tweet in the pretraining dataset is 11, whereas e.g. the board posts from the Internet Argument Corpus version 1 (Oraby et al., 2016) has an average of 66 tokens with some observations being much longer.

5 Model Analysis

5.1 Importance of emoji diversity

One of the major differences between this work compared to previous papers using distant supervision is the diversity of the noisy labels used (see §2). For instance, both Deriu et al. (2016) and Tang et al. (2014) only used positive and negative emoticons as noisy labels. Other instances of previous work have used slightly more nuanced sets of noisy labels (see §2), but to our knowledge our set of noisy labels is the most diverse yet. To analyze the effect of using a diverse emoji set we create a subset of our pretraining data containing tweets with one of 8 emojis that are similar to the positive/negative emoticons used by Tang et al. (2014) and Hu et al. (2013) (the set of emoticons and corresponding emojis are available in the supplemental material). As the dataset based on this reduced set of emojis contains 433M tweets, any difference in performance on benchmark datasets is likely linked to the diversity of labels rather than differences in dataset sizes.

We train our DeepMoji model to predict whether the tweets contain a positive or negative emoji and evaluate this pretrained model across the benchmark datasets. We refer to the model trained on the subset of emojis as *DeepMoji-PosNeg* (as opposed to *DeepMoji*). To test the emotional representations learned by the two pretrained models the ‘last’ transfer learning approach is used for the comparison, thereby only allowing the models to map already learned features to classes in the target dataset. Table 6 shows that DeepMoji-PosNeg yields lower performance compared to DeepMoji across all 8 benchmarks, thereby showing that the diversity of our emoji types encourage the model to learn a richer representation of emotional content in text that is more useful for transfer learning.

Many of the emojis carry similar emotional

Table 6: 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.

| 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 |

content, but have subtle differences in usage that our model is able to capture. Through hierarchical clustering on the correlation matrix of the DeepMoji model’s predictions on the test set we can see that the model captures many similarities that one would intuitively expect (see Figure 3). For instance, the model groups emojis into overall categories associated with e.g. negativity, positivity or love. Similarly, the model learns to differentiate within these categories, mapping sad emojis in one subcategory of negativity, annoyed in another subcategory and angry in a third one.

5.2 Model architecture

Our DeepMoji model architecture as described in §3.2 use an attention mechanism and skip-connections to ease the transfer of the learned representation to new domains and tasks. Here we compare the DeepMoji model architecture to that of a standard 2-layer LSTM, both compared using the ‘last’ transfer learning approach. We use the same regularization and training parameters.

As seen in Table 6 the DeepMoji model performs better than a standard 2-layer LSTM across all benchmark datasets. The two architectures performed equally on the pretraining task, suggesting that while the DeepMoji model architecture is indeed better for transfer learning, it may not necessarily be better for single supervised classification task with ample available data.

A reasonable conjecture is that the improved transfer learning performance is due to two factors: a) the attention mechanism with skip-connections provide easy access to learned low-

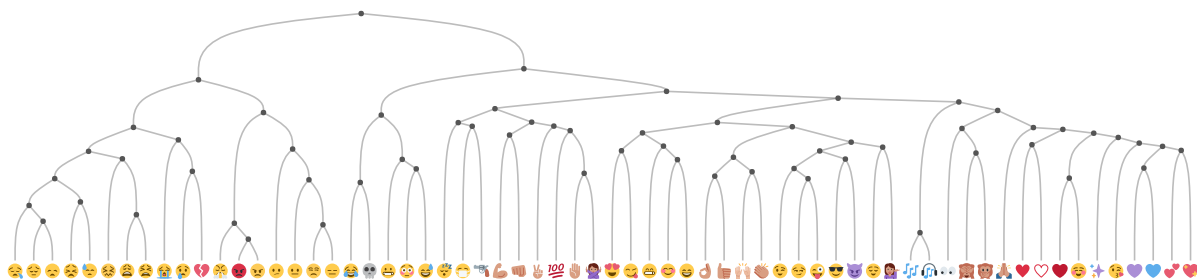


Figure 3: Hierarchical clustering of the DeepMoji model’s predictions across categories on the test set. The **dendrogram** shows how the model learns to group emojis into overall categories and subcategories based on emotional content. **The y-axis is the distance on the correlation matrix of the model’s predictions** measured using average linkage. More details are available in the supplementary material.

level features for any time step, making it easy to use this information if needed for a new task b) the improved gradient-flow from the output layer to the early layers in the network due to skip-connections (Graves, 2013) is important when adjusting parameters in early layers as part of transfer learning to small datasets. Detailed analysis of whether these factors actually explain why our architecture outperform a standard 2-layer LSTM is left for future work.

5.3 Analyzing the effect of pretraining

Performance on the target task benefits strongly from pretraining as shown in Table 5 by comparing DeepMoji (new) to DeepMoji (chain-thaw). In this section we experimentally decompose the benefit of pretraining into 2 effects: word coverage and phrase coverage. These two effects help regularize the model by preventing overfitting (see the supplementary details for an visualization of the effect of this regularization).

There are numerous ways to express a specific sentiment, emotion or sarcastic comment. Consequently, the test set may contain specific language use not present in the training set. The pretraining helps the target task models attend to low-support evidence by having previously observed similar usage in the pretraining dataset. We first examine this effect by measuring the improvement in word coverage on the test set when using the pretraining with word coverage being defined as the % of words in the test dataset seen in the training/pretraining dataset (see Table 7). An important reason why the ‘chain-thaw’ approach outperforms other transfer learning approaches is can be used to tune the embedding layer with limited risk of overfitting. Table 7 shows the increased word

coverage from adding new words to the vocabulary as part of that tuning.

Note that word coverage can be a misleading metric in this context as for many of these small datasets a word will often occur only once in the training set. In contrast, all of the words in the pretraining vocabulary are present in thousands (if not millions) of observations in the emoji pretraining dataset thus making it possible for the model to learn a good representation of the emotional and semantic meaning. The added benefit of pretraining for learning word representations therefore likely extends beyond the differences seen in Table 7.

Table 7: 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’).

| 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% |

To examine the importance of capturing phrases and the context of each word, we evaluate the accuracy on the SS-Youtube dataset using a fastText classifier pretrained on the same emoji dataset as our DeepMoji model. This fastText classifier is almost identical to only using the embedding layer

from the DeepMoji model. We evaluate the representations learned by fine-tuning the models as feature extractors (i.e. using the ‘last’ transfer learning approach). The fastText model achieves an accuracy of 63% as compared to 93% for our DeepMoji model, thereby emphasizing the importance of phrase coverage. One concept that the LSTM layers likely learn is negation, which is known to be important for sentiment analysis (Wiegand et al., 2010).

5.4 Comparing with human-level agreement

To understand how well our DeepMoji classifier performs compared to humans, we created a new dataset of random tweets annotated for sentiment. Each tweet was annotated by a minimum of 10 English-speaking Amazon Mechanical Turkers (MTurk’s) living in USA. Tweets were rated on a scale from 1 to 9 with a ‘Do not know’ option, and guidelines regarding how to rate the tweets were provided to the human raters. The tweets were selected to contain only English text, no mentions and no URL’s to make it possible to rate them without any additional contextual information. Tweets where more than half of the evaluators chose ‘Do not know’ were removed (98 tweets).

For each tweet, we select a MTurk rating random to be the ‘human evaluation’, and average over the remaining nine MTurk ratings are averaged to form 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 that the label categories are clearly separated, we removed neutral tweets in the interval [4.5, 5.5] (roughly 29% of the tweets). The remaining dataset consists of 7 347 tweets. Of these tweets, 5000 are used for training/validation and the remaining are used as the test set. Our DeepMoji model is trained using the chain-thaw transfer learning approach.

Table 8 shows that the agreement of the random MTurk rater is 76.1%, meaning that the randomly selected rater will agree with the average of the nine other MTurk-ratings of the tweet’s polarity 76.1% of the time. Our DeepMoji model achieves 82.4% agreement, which means it is better at capturing the average human sentiment-rating than a single MTurk rater.

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

| | Agreement |
|----------|--------------|
| Random | 50.1% |
| fastText | 71.0% |
| MTurk | 76.1% |
| DeepMoji | 82.4% |

6 Conclusion

We have shown how the millions of texts on social media with emojis can be used for pretraining models, thereby allowing them to learn representations of emotional content in texts. Through comparison with an identical model pretrained on a subset of emojis, we find that the diversity of our emoji set is important for the performance of our method. We release our pretrained DeepMoji model with the hope that other researchers will find good use of them for various emotion-related NLP tasks⁴.

Acknowledgments

The authors would like to thank Janys Analytics for generously allowing us to use their dataset of human-rated tweets and the associated code to analyze it. Furthermore, we would like to thank Max Lever, who helped design the website, and Han Thi Nguyen, who helped code the software that is provided alongside the pretrained model.

References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations (ICLR)*.
- Yoshua Bengio et al. 2012. Deep learning of representations for unsupervised and transfer learning. In *29th International Conference on Machine Learning (ICML) – Workshop on Unsupervised and Transfer Learning*, volume 27, pages 17–36.
- Sven Buechel and Udo Hahn. 2016. Emotion analysis as a regression problem - dimensional models and their implications on emotion representation and metrical evaluation. In *22nd European Conference on Artificial Intelligence (ECAI)*.

⁴Available with preprocessing code, examples of usage, benchmark datasets etc. at github.com/bfelbo/deepmoji

- François Chollet et al. 2015. Keras. <https://github.com/fchollet/keras>.
- Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *25th International Conference on Machine Learning (ICML)*, pages 160–167.
- Jan Deriu, Maurice Gonzenbach, Fatih Uzdilli, Aurelien Lucchi, Valeria De Luca, and Martin Jaggi. 2016. Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks with distant supervision. *Proceedings of SemEval*, pages 1124–1128.
- Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. Decaf: A deep convolutional activation feature for generic visual recognition. In *31th International Conference on Machine Learning (ICML)*, volume 32, pages 647–655.
- Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description. In *4th International Workshop on Natural Language Processing for Social Media (SocialNLP)*.
- Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research (JMLR)*, 11:625–660.
- Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *30th Conference on Neural Information Processing Systems (NIPS)*, pages 1019–1027.
- Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12).
- Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.
- Xia Hu, Jiliang Tang, Huiji Gao, and Huan Liu. 2013. Unsupervised sentiment analysis with emotional signals. In *Proceedings of the 22nd international conference on World Wide Web (WWW)*, pages 607–618. ACM.
- Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are word embedding-based features useful for sarcasm detection? In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.
- Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations (ICLR)*.
- FA Kunneman, CC Liebrecht, and APJ van den Bosch. 2014. The (un)predictability of emotional hashtags in twitter. In *52th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *27th Conference on Neural Information Processing Systems (NIPS)*, pages 3111–3119.
- Saif Mohammad. 2012. #emotional tweets. In *The First Joint Conference on Lexical and Computational Semantics (*SEM)*, pages 246–255. Association for Computational Linguistics.
- Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. 2016. Semeval-2016 task 4: Sentiment analysis in twitter. In *10th International Workshop on Semantic Evaluation (SemEval)*, pages 1–18.
- Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn Walker. 2016. Creating and characterizing a diverse corpus of sarcasm in dialogue. In *17th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, page 31.
- Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.
- Jonathon Read. 2005. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *ACL student research workshop*, pages 43–48. Association for Computational Linguistics.
- Hassan Saif, Miriam Fernandez, Yulan He, and Harith Alani. 2013. Evaluation datasets for twitter sentiment analysis: a survey and a new dataset, the sts-gold. In *Workshop: Emotion and Sentiment in Social and Expressive Media: approaches and perspectives from AI (ESSEM) at AI*IA Conference*.
- Valentina Sintsova, Claudiu-Cristian Musat, and Pearl Pu. 2013. Fine-grained emotion recognition in olympic tweets based on human computation. In *4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis (WASSA)*.
- Jonas Sjöberg and Lennart Ljung. 1995. Overtraining, regularization and searching for a minimum, with application to neural networks. *International Journal of Control*, 62(6):1391–1407.

- Jacopo Staiano and Marco Guerini. 2014. Depechemood: A lexicon for emotion analysis from crowd-annotated news. In *52th Annual Meeting of the Association for Computational Linguistics (ACL)*. Association for Computational Linguistics.
- Carlo Strapparava and Rada Mihalcea. 2007. Semeval-2007 task 14: Affective text. In *4th International Workshop on Semantic Evaluations (SemEval)*, pages 70–74. Association for Computational Linguistics.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *28th Conference on Neural Information Processing Systems (NIPS)*, pages 3104–3112.
- Jared Suttles and Nancy Ide. 2013. Distant supervision for emotion classification with discrete binary values. In *International Conference on Intelligent Text Processing and Computational Linguistics (CI-Ling)*, pages 121–136. Springer.
- Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *52th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1555–1565.
- Theano Development Team. 2016. [Theano: A Python framework for fast computation of mathematical expressions](#). *arXiv e-prints*, abs/1605.02688.
- Mike Thelwall, Kevan Buckley, and Georgios Paltoglou. 2012. Sentiment strength detection for the social web. *Journal of the American Society for Information Science and Technology (JASIST)*, 63(1):163–173.
- Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. 2010. Sentiment strength detection in short informal text. *Journal of the American Society for Information Science and Technology*, 61(12):2544–2558.
- Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. 2012. A corpus for research on deliberation and debate. In *International Conference on Language Resources and Evaluation (LREC)*, pages 812–817.
- Harald G Wallbott and Klaus R Scherer. 1986. How universal and specific is emotional experience? evidence from 27 countries on five continents. *International Social Science Council*, 25(4):763–795.
- Michael Wiegand, Alexandra Balahur, Benjamin Roth, Dietrich Klakow, and Andrés Montoyo. 2010. A survey on the role of negation in sentiment analysis. In *Workshop on Negation and Speculation in Natural Language Processing (NeSp-NLP)*, pages 60–68. Association for Computational Linguistics.
- Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alexander J Smola, and Eduard H Hovy. 2016. Hierarchical attention networks for document classification. In *HLT-NAACL*.