

Multi-task Emoji Learning

Francesco Barbieri[♣] Lu s Marujo[ ] Pradeep Karuturi[ ] William Brendel[ ]

[♣] Large Scale Text Understanding Systems Lab, TALN, UPF, Barcelona, Spain

[ ] Snap Inc. Research, Venice, California, USA

[♣]{name.surname}@upf.edu, [ ]{name.surname}@snap.com






Abstract

Emojis are very common in social media and understanding their underlying semantics is of great interest from a Natural Language Processing point of view. In this work, we investigate emoji prediction in short text messages using a multi-task pipeline that simultaneously predicts emojis, their categories and sub-categories. The categories are either manually predefined in the unicode standard or automatically obtained by clustering over word embeddings. We show that using this categorical information adds meaningful information, thus improving the performance of emoji prediction task. We systematically analyze the performance of the emoji prediction task by varying the number of training samples and also do a qualitative analysis by using attention weights from the prediction task.

1 Introduction

Emojis are a popular set of ideograms created in the late 1990s to enrich written communication by adding nonverbal expressive power to digital communication. These symbols can be used by human readers to convey emotions and information in a condensed form. As Snapchat, Twitter and other social media platforms have become popular, so has the usage of emojis.

Despite their popularity, there is very little research work in predicting emojis.

Over the past few years, the interest in emoji research has increased and several studies have been published in the area of distributional semantics [BRS16, ERA⁺16, WBSD17b, WBSD17a, BCC18], sentiment analysis [NSSM15, HGS⁺17, KK17, RPG⁺18] and multimodal systems [CMS15, CSG⁺18, BBS18]. In the past year researchers also focused on the possibility of predicting emojis in a text message [BBS17, FMS⁺17]. The emoji prediction task consists in predicting the original emoji present in a tweet (or snap caption) given only the non-emoji textual content. Prior explorations of emoji prediction tended to focus on less than 2% of the total number (2653) of emojis in Unicode 6 standard¹ emojis. Another limitation of those papers was that emoji prediction could be ambiguous. For example, when the model predicts the correct label to be , emojis like , , , or  can also be valid predictions.

In this work, we extended the emoji prediction task to 300 emojis in order to study a larger number of emojis along with their unicode standard categories, sub-categories, and the new semantic clusters that we created. We are not aware of any previous research work focused on either predicting a large number of emojis (300), or using a multi-task approach to predict emojis or emoji categories. We also do a systematic analysis of how the number of training samples affect the performance of the emoji prediction task. To mitigate the problem of emoji ambiguity, we concentrate on broad emoji category prediction in addition to that of individual emoji prediction. We grouped emojis in two different ways. The first one was defined by the Unicode consortium², which groups emojis into seven categories (e.g., “Smileys & People”, “Nature”) and 74 sub-categories (e.g., “face-positive”, “face-negative”). The main categories are commonly found on mobile phone keyboards as shown in Figure 1). Alternatively

Copyright   2018 held by the author(s). Copying permitted for private and academic purposes.

In: S. Wijeratne, E. Kiciman, H. Saggion, A. Sheth (eds.): Proceedings of the 1st International Workshop on Emoji Understanding and Applications in Social Media (Emoji2018), Stanford, CA, USA, 25-JUN-2018, published at <http://ceur-ws.org>

¹www.unicode.org/emoji/charts/full-emoji-list.html

²www.unicode.org/emoji/charts/emoji-ordering.html



Figure 1: Screenshot of Apple’ Emojis keyboard including a subset of Smileys & People category emojis, and buttons to access the remaining categories.

we also created semantic clusters using embeddings.

We use a Multi-task approach to combine the tasks of emoji and category prediction. Multi-task approaches [Car, Car97, Ben12, CW08] improves generalization by transferring information across different tasks and improving each task individually. In particular, multi-task learning with simultaneous training on multiple tasks has demonstrated promising results [CW08][FMS⁺17][Ben12].

Our work performs multi-task learning by training a single model with multiple outputs (the dataset is annotated with multiple labels) and we evaluate using our gold standard created from Twitter and Snap public posts as described in the Datasets section. The subjectivity of emoji interpretation makes emoji prediction a very challenging task. Nevertheless our work shows that simultaneously predicting emojis, their categories, and sub-categories in a multitask framework improves the overall results. It not only improves emoji prediction, but it also helps with the identification of emoji categories, which can be particularly more relevant when the emoji prediction model is less precise.

The remainder of this work is organized in the following way: The next section describe the datasets used in our experiments. We then present the Deep Learning Models explored to solve our research problem, Finally, we discuss the experiments, results and then conclude with future research directions.

2 Datasets

In this study we explore emoji prediction for two different datasets: Twitter and Snapchat captions. We select documents (*tweets* and *snaps*) that contain a single emoji, and at least three tokens apart from the emoji. We restrict to documents containing a single emoji so as to minimize the interference of the presence of other emojis in the emoji prediction task. We also consider only the documents that include the most frequent 300 emojis in each dataset. We restrict to the top 300 emojis only due to lack of meaningful number of examples beyond that. A subset of the most frequent emojis for each dataset is reported in Table 1.

Note that we remove skin color from the emojis³ to avoid generating very similar labels. Table 2 includes statistics on the two datasets. We can see that Snap captions are shorter than tweets, while average word length is similar. Another important difference between the two datasets is the most frequent emojis used. Table 1 shows the 60 most frequent emojis in each dataset (on the top Twitter and on the bottom Snap data), along with the number of documents that include each emoji. In both datasets the frequency is very unbalanced: 16% of tweets and 25% of snaps include one of the three most frequent emojis ❤️, 😊, 🙄. Therefore we use a balanced dataset in our experiments, in order to give same importance to each emoji, independent of the frequency of usage. We subsample the most frequent emojis in order to match the same number of examples of the least represented emoji (1,500 examples for Twitter and 3,000 for Snap data). We show that using fewer than 1,500 examples per emoji leads to a drastic decrease in accuracy of the emoji detection (see Figure 3). We focus our experiments on 300 emojis because we do not have more than 1,500 tweets per emoji beyond the top 300 emojis in our Twitter dataset. For our experiments we randomly chose 80% of the documents for training, 10% for validation and 10% for testing.

2.1 Twitter Dataset

The Twitter dataset contains 50 million tweets retrieved using Twitter API. Tweets were posted between January 2016 and April 2017 and were geolocalized in the United States. We removed hyperlinks from each tweet, and lowercased all textual content in order to reduce noise and sparsity. Since Twitter data includes a large percentage of bot data, we filter noise as much as possible, removing repeated tweets (or very similar ones) and selected a maximum of five tweets per user. From this dataset, we selected tweets including anyone of the 300 most frequently occurring emojis and at least three tokens (without the emoji), resulting in a final dataset composed of 2,862,505 tweets.

2.2 SnapCaption

SnapCaption is an in-house Snapchat internal dataset containing only Snapchat *captions*. A caption is the textual overlay component of a snap. These captions were collected exclusively from snaps submitted to public and crowd-sourced stories (as known as *Live Stories* or *Our Stories*). Examples of such public crowd-sourced stories are “New York Story” or “Thanksgiving Story”. All captions were posted in one year period and do not contain any image or any

³E.g., 🍌 🍌 🍌 🍌 🍌 are mapped to one single label 🍌

Table 1: 60 most frequent emoji for the Twitter (top) and Snap (bottom) datasets.

❤️	😂	💕	🔥	😄	😏	🌟	💙	👉	😘	📷	🇺🇸	☀️	👉	👉	🙏	💜	😘	🏆	😘	🌲	📷	💪	😘	🌈	😘	💋	❄️	😘	
235	112	111	63	52	50	45	41	38	36	35	33	31	31	31	29	29	28	28	27	27	27	27	26	24	24	23	23	23	
👉	💕	😘	🎵	👉	😘	💕	👉	😘	😘	🏆	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	
23	22	20	19	19	18	18	18	17	17	16	16	16	16	16	15	15	15	15	15	15	14	14	14	14	13	13	13	12	12
<hr/>																													
😂	😘	❤️	😘	👉	👉	🔥	😘	😘	🙏	💕	😘	😘	😘	😘	💙	🏆	😘	🌈	👉	💪	😘	🌲	👉	😘	😘	😘	👉	👉	
3343	2425	1734	645	617	578	507	433	418	395	391	380	375	364	356	321	315	313	278	273	267	266	234	226	226	225	222	219	212	197
👉	😘	👉	💕	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	👉	
<hr/>																													
194	194	191	188	187	186	186	181	174	170	169	167	161	157	154	153	153	150	148	147	147	145	139	136	136	134	134	133	124	123

other associated information. This dataset contains 30,004,519 captions.

2.3 Categories and Clusters of Emojis

We also consider broader classes of emojis, such as unicode categories and semantic clusters. The unicode consortium defines a set of 7 categories categories and 74 sub-categories.

The problem with Unicode categories and sub-categories is that they fail to accurately capture semantically related emojis. Emojis like 😊 and 😄 are both in the sub-category neutral faces even though they clearly indicate different emotions. Another example is ❤️ and 😊 that are semantically similar, but they appear in different categories (“Smiling Faces” and “Emotions”) even though they address nearly identical meanings. To overcome this limitation, we propose a second approach to automatically organize emojis by clustering them using pre-trained word embeddings similar to emoji2vec [?]. These clusters have the advantage of better capturing the semantic information of emojis. For example ❤️ and 😊 are in the same cluster. These clusters are an important aspect to consider because they are based on how emojis co-occur in short text messages from tweets and captions of public snaps. We pretrained two different sets of skip-gram embeddings [MLS13] for Twitter and Snap. The first skip-gram model was trained on a dataset of about 70 million tweets and the second skip-gram model was trained on about 100 million Snap captions. Using the embeddings of the 300 most frequent emojis of each dataset, we created two sets of 30 clusters using a k-means algorithm. The number of clusters was defined based on qualitative analysis (clusters that seemed to better organize emojis by semantics). In addition, the number of clusters was selected such that each cluster has a similar number of emojis that are usually displayed on a mobile keyword. As a result, we would be able to just provide an icon to access directly each cluster in a similar way as the Figure 1 shows for the top categories. The resulting clusters will group semantically similar emojis (like in [BKRS16] where 11 cluster are created for 100 emojis), grouping love, sad faces, hand/gestures, animals, food, drinks, parties, Christmas, and so on.

Table 2: Average, standard deviation and Median length of words and characters of the two datasets.

Dataset	Words			Chars		
	Avg.	Std.	Median	Avg.	Std.	Median
Twitter	12.73	4.23	12	91.14	24.67	92
Snap	5.01	2.29	4	25.62	11.75	23

3 Models

Our main architecture, illustrated in Fig. (2), starts with our character and word embedding modules whose outputs are fused by our feature attention unit and the word attention unit. Finally the fully connected layers and the softmax play the role of the final multi-task classifier.

Previous approaches [BBS17, FMS⁺17] have successfully learned LSTM models for emoji prediction tasks. We experimented different plain LSTMs, stacked LSTMs [FMS⁺17], and different word representations before solidifying on our final model architecture Fig. (2). In addition, we explored single task models and multi-task models. In the case of the multi-task models, the entire network is shared and the specialization only occurs at the final stage to predict specific labels of each task. This specialization is accomplished through specific linear transformations. Finally we used a cross entropy loss function for all classification tasks. In the case of multitask learning, the final loss is the sum of each single loss⁴. In the following subsections, we detail each stage of our main architecture.

3.1 Word Representation

The word embeddings are learned together with the updates to the model. For out-of-vocabulary words (OOVWs), we used a fixed representation that is handled as a separate word. In order to train the fixed representation for OOVWs, we stochastically replace (with $p = 0.5$) each word that occurs only once in the training data. When we use pre-trained word embeddings, that are concatenated with the learned vector.

⁴We also experimented weighted sum, with various weights, but the best results are obtained with a simple sum of the losses.

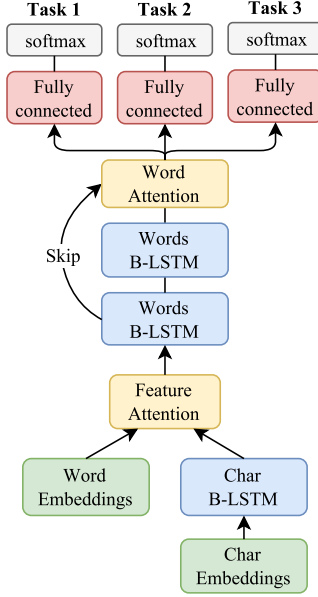


Figure 2: Final architecture of our model

3.2 Char Representation

In addition, we use a character based embedding [LLM⁺15, ?] stacked with a B-LSTM [GS05], producing a character-based word embedding that focuses on word spelling variants. Indeed, the character-based word embedding learned similar representations for words that are orthographically similar, and thus are expected to handle different alternatives of the same word types that normally occur in social media.

3.3 Bi-directional LSTMs

Our bi-directional LSTM modules, [GS05] named B-LSTM in Fig. (2), consists of a forward LSTM that processes an input message from left to right, while the backward LSTM processes it in the reverse direction. As a result, the message representation \mathbf{s} is based on both the forward and backward LSTM encoding:

$$\mathbf{s} = \max \{ \mathbf{0}, \mathbf{W}[\mathbf{h}_{\mathbf{fw}}; \mathbf{h}_{\mathbf{bw}}] + \mathbf{d} \}$$

where \mathbf{W} is a learned parameter matrix, \mathbf{fw} is the forward LSTM encoding of the message, \mathbf{bw} is the backward LSTM encoding of the message, and \mathbf{d} is a bias term, and we use a component-wise ReLU as the non-linear unit. We use B-LSTM modules for both word and sentence representations, namely Char B-LSTM and Words B-LSTMs in our architecture Fig. (2). Char B-LSTM takes a sequence of characters and outputs a word embedding vector. This output is mixed with another word representation via our feature attention module. Then, the stacked Words LSTMs receive sequences of word representations from the attention module, and output sentence embedding vectors.

3.4 Feature Attention

The feature attention module aims to linearly fuse multiple input signals instead of simply concatenating them. In our architecture, this module learns a unified word representation space, i.e. it produces a single vector representation with aggregated knowledge among our multiple input word representations, based on their weighted importance. We can motivate this module from the following observations.

Prior work, [BBS17] combines both word representation $\mathbf{x}^{(w)}$ and character-level representation $\mathbf{x}^{(c)}$ by simply concatenating the word and character embeddings at each LSTM decoding step $\mathbf{h}_t = \text{LSTM}([\mathbf{x}_t^{(w)}; \mathbf{x}_t^{(c)}])$. However, this naive concatenation results in inaccurate decoding, specifically for unknown word token embeddings, e.g., an all-zero vector $\mathbf{x}_t^{(w)} = \mathbf{0}$ or a random vector $\mathbf{x}_t^{(w)} = \epsilon \sim U(-\sigma, +\sigma)$, or even for out-of-vocabulary words. While this concatenation approach does not cause significant errors for well-formatted text, we observe that it induces performance degradation for our social media post datasets which contain a significant number of slang words, i.e., misspelled or out-of-vocabulary words. As a result, we use a feature attention module, that adaptively emphasizes each feature representation in a global manner at each decoding step t . This process produces a soft-attended context vector $\bar{\mathbf{x}}_t$ as an input token for the next stacked B-LSTM that takes care of the sentences embedding. [RCP16] introduced a similar approach, where the character embeddings are weighted with an attention module. We use the following method:

$$\begin{aligned} [\mathbf{a}_t^{(w)}, \mathbf{a}_t^{(c)}] &= \sigma(\mathbf{W}_m \cdot [\mathbf{x}_t^{(w)}; \mathbf{x}_t^{(c)}] + \mathbf{b}_m) \\ \alpha_t^{(m)} &= \frac{\exp(\mathbf{a}_t^{(m)})}{\sum_{m' \in \{w, c\}} \exp(\mathbf{a}_t^{(m')})} \quad \forall m \in \{w, c\} \end{aligned} \quad (1)$$

where $\alpha_t = [\alpha_t^{(w)}; \alpha_t^{(c)}] \in \mathbb{R}^2$ is an attention vector at each decoding step t , and $\bar{\mathbf{x}}_t$ is a final context vector at t that maximizes information gain for \mathbf{x}_t . Note that this feature attention requires each feature representation to have the same dimension (e.g. $\mathbf{x}_t^{(w)}, \mathbf{x}_t^{(c)} \in \mathbb{R}^p$), and that the transformation via \mathbf{W}_m essentially enforces each feature representation to be mapped into the same unified subspace, with the output of the transformation encoding weighted discriminative features for classification of emojis.

3.5 Word Attention

Not all the words have the same importance in the representation of a document. We use the attention

mechanism introduced in [YYD⁺16]:

$$\begin{aligned} u_{it} &= \tanh(W_w h_{it} + b_w) \\ \alpha_{it} &= \frac{\exp(u_{it}^\top u_w)}{\sum_t \exp(u_{it}^\top u_w)} ; d_i = \sum_t \alpha_{it} h_{it} \end{aligned} \quad (2)$$

where the final document representation d_i is a weighted average of the hidden representation h_{it} of the LSTM. The weights α_{it} are learned by the use of a Multi-Layer Perceptron (linear transformation W and biases b) with \tanh as non-linear operation, and a softmax to compute the probability of each word.

4 Experiments And Results

We use two main variations for experiments: **Single-Task Prediction** of emojis, unicode categories, and emoji clusters, and **Multi-Task Prediction**, where we combine the single tasks in one single model. We also evaluate the impact of our different modules including the combination of word/char LSTMs and the word attention unit. Finally we investigate the influence of the number of layers for the LSTMs.

4.1 Single-Task Prediction

We explore three different tasks: (i) the emoji prediction task proposed by [BBS17], (ii) prediction of unicode emoji categories (the emoji in the text belong to the faces, animal, objects) and sub-categories (positive faces, animal-mammal), and (iii) prediction of automatic clusters that we previously generated using pre-trained word embeddings.

4.1.1 Predicting Emojis

Given a set of documents, each document containing only one emoji class, the task consists of predicting the emoji from the text. For this task, we tested the influence of the number of emoji classes and the number of examples per class. More precisely, for each experiment, we extract a balanced dataset of N_{class} emoji classes, and N_{data} examples per class, with $N_{class} = \{20, 50, 100, 200, 300\}$ and $N_{data} = \{100, 500, 1000, 1500, 2000, 2500, 3000\}$. N_{class} and N_{data} are tested independently: when we vary N_{class} , we fix N_{data} to 3000, and when we vary N_{data} we fix N_{class} to 300. Figure 3 shows our experiments with the Snapchat dataset. It is clear that using more examples per class improves our model by around 1% absolute point from 1500 to 3000 examples. For >2000 examples the system converges to its optimum.

From Figure 4, we observe that Twitter data is easier to model than Snap data. In the 300 emoji prediction task the best accuracy at top 5 (a@5) on Twitter data is 40.05% while on Snap data it is 34.25% (see

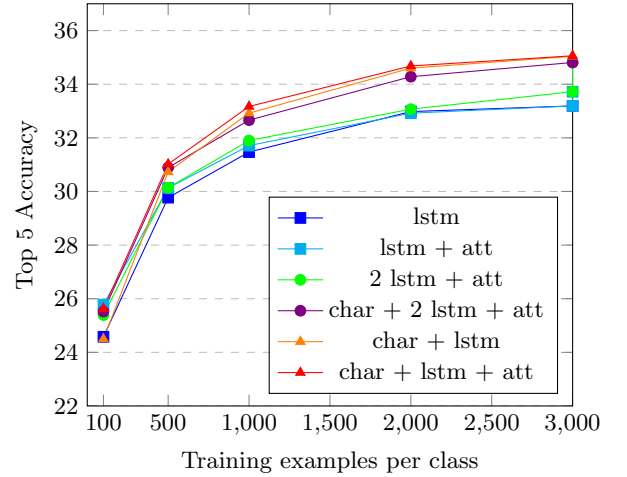


Figure 3: Acc@top 5 of the same algo. but variable nr. of training instances per class (from 100 to 3000 examples for each emoji) on SnapCaptions. Test and validation set are fixed for each experiment.

Table 3). There are several reasons that could explain this difference in results. One reason is the length of the text messages, since in Twitter there are on average twelve words per message, while on Snap has only five (see Table 2). Another reason could be the missing visual context of Snap posts⁵, while only a small percentage of tweets is complemented with a visual content. For this reason, tweets contain typically less semantic ambiguity.

Table 3 highlights the best performing systems on the emoji prediction task. For the two datasets state of the art systems are outperformed by the combination of additional components. For example, adding a word attention module improves the baseline of [BBS17]. Finally, there is an important difference when predicting 20 and 300 emojis. We plot on the left of Figure 3 the accuracy of same model architecture (Char + 2 LSTM + word attention) on the emoji prediction task for different numbers of emojis (20 to 300). Best accuracy at top 5 (a@5) drops from 20 to 100, and then remains constant. We observe the same drop using F1 (that only considers whether an emoji is predicted as first option), however, having more than 100 classes results in improvement. This is probably due the type of the more rare emoji classes added after the most 100 frequent ones, that are more specific (like 🍌, 🚩, or 🍷) hence easier to predict.

4.1.2 Predicting Unicode Emoji Categories and Sub-categories

We predict Unicode emojis categories and sub-categories using the text message that includes only

⁵Snap text messages are captions of images and videos posted by Snapchat users, see Datasets section.

Table 3: Emoji prediction results using multiple number of emojis (20 to 300) and different models. We use the original implementation of [FMS⁺17], while we implement [BBS17].

Dataset	Models	20		50		100		200		300	
		F1	a@5	F1	a@5	F1	a@5	F1	a@5	F1	a@5
Twitter	LSTM	25.32	62.17	19.51	46.69	18.03	40.49	19.44	39.11	19.86	38.44
	LSTM + Att.	25.51	62.82	19.49	47.21	18.3	40.56	19.61	39.10	19.38	38.41
	2 LSTM + Att. [FMS ⁺ 17]	24.47	62.38	19.64	47.01	18.36	40.60	19.60	39.09	20.09	38.70
	Char + LSTM [BBS17]	26.81	63.37	20.21	47.83	18.87	41.41	20.49	40.14	21.27	40.06
	Char + LSTM + Att.	27.37	64.33	20.91	48.23	18.88	41.8	21.19	40.65	21.59	40.06
	Char + 2 LSTM + Att.	26.85	64.12	20.36	48.51	18.82	41.92	20.66	40.51	20.53	39.22
Snap	LSTM	25.46	53.51	18.62	43.96	15.06	34.72	16.08	32.96	17.57	32.44
	LSTM + Att.	25.58	53.67	18.86	44.44	15.40	35.20	16.47	32.95	17.46	32.63
	2 LSTM + Att. [FMS ⁺ 17]	24.30	53.01	18.64	43.59	15.40	35.03	16.73	33.26	18.07	32.94
	Char + LSTM [BBS17]	24.84	53.37	19.26	44.75	15.50	35.38	17.39	33.89	18.80	33.98
	Char + LSTM + Att.	26.01	54.34	19.39	45.1	15.58	35.61	17.44	34.26	18.64	33.86
	Char + 2 LSTM + Att.	25.72	53.81	18.95	45.05	16.18	36.03	17.51	33.97	18.87	34.25

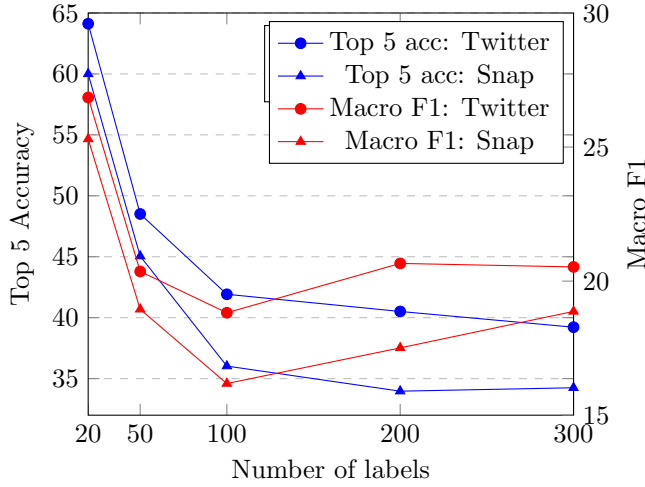


Figure 4: F1 and Acc. @ top 5 for the model “Char + 2 LSTM + Word Att.” on Twitter and Snap data.

one emoji as we did in the emoji prediction task.

Table 4 shows the prediction results using macro-F1 and a@5 evaluation metrics. In the first two blocks (main and sub lines), we predict the main category and sub-category respectively. The third block details the clusters’ evaluation results, and the last block presents the emoji prediction results. In the first line of each block are the single-task results and the remaining lines include the ones using a multi-task framework.

4.1.3 Predicting Clusters

Given a text message containing an emoji e we predict the cluster that emoji e belongs to. Cluster creation is described in the dataset section. Cluster results are reported in Table 4, in the lines corresponding to “Semantic Clusters”. The results are better on Snap than Twitter for broader classes and our clusters capture better semantics than the categories and

Table 4: Results for single and multi-task prediction of emojis including *main* unicode categories, *sub*-categories, and clusters.

Pred. Task	Loss	Twitter		Snap	
		F1	A@5	F1	A@5
Main Category	Main	46.56	84.70	45.23	87.90
	Main + Sub	48.34	85.87	45.07	87.79
	Main + Emoji	48.17	85.52	44.54	87.83
	Main + sub + Emoji	48.52	85.90	44.64	95.52
Sub Category	Sub	31.62	51.02	32.15	53.81
	Sub + Main	31.84	51.86	31.72	53.43
	Sub + Emoji	32.00	52.23	31.99	53.72
	Sub + Main + Emoji	32.24	52.40	31.88	65.19
Semantic Clusters	Clusters	34.10	53.56	34.77	53.22
	Clusters + Emoji	35.42	55.90	34.90	53.64
Emoji	Emoji	21.59	40.06	18.64	33.86
	Emoji + Main	21.62	37.80	19.05	34.24
	Emoji + Sub	21.58	37.91	18.75	34.27
	Emoji + Main + Sub	21.44	37.81	18.78	34.05
	Emoji + Clusters	21.30	37.90	19.05	29.78

sub-categories of Unicode Standard.

4.2 Multi-Task Predictions

In Table 4 we show the multi-task prediction results. We considered multiple multi-task combinations. Learning more than one objective task simultaneously helps in the main category prediction, as macro F1 improves from 46.56% to 48.52% (4.2% relative improvement) when adding also sub-category and emoji losses. Sub-categories prediction also improves when it is learned together with main categories and emojis.

On Snap data, category and sub-category prediction tasks do not improve using a multitask approach in terms of macro F1, but we obtain a relative improvement of 8.67% and 21.14% using a@5.

The clusters prediction tasks also benefit from multi-task learning when combined with the emoji prediction. However, emoji prediction seems not to improve much in a multi-task setting for Twitter. Emoji

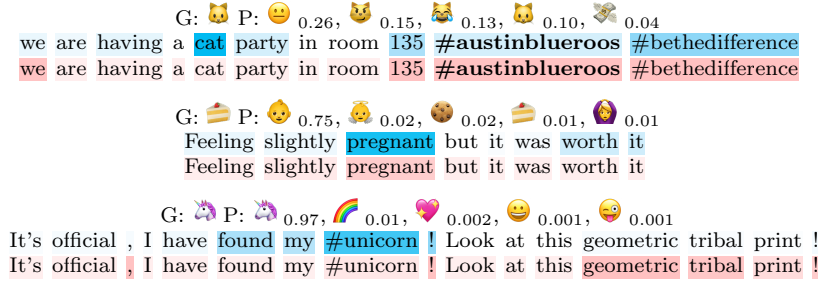


Figure 5: Word and Feature attention visualization. The first line highlights in blue word attention, while the second line shows the feature attention. Uncolored words mean almost zero attention over them.

Table 5: Top and bottom 10 emojis with best accuracy on the Twitter (top) and on Snap (bottom).

🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷
87.5	83.87	83.53	83.33	81.33	80.88	79.86	79.86	79.56	78.95
🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷
4.71	4.57	4.43	3.97	1.94	1.94	1.52	1.24	0.63	0.58
🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷
89.82	78.27	77.19	76.15	75.72	74.1	73.48	73.03	71.5	71.46
🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷	🍷
3.46	3.37	3.13	2.92	2.82	2.41	1.86	1.01	0.29	0

prediction on Snap improves from 33.86% to 34.27% or 1.21% relative improvement in terms of a@5 when it is learned together with Unicode sub-categories.

4.3 Qualitative Analysis

We analyzed in detail our emoji prediction approach (char + 2 LSTM + attention) based on the best performing system described in the previous section. This analysis enumerates the emojis that are easier and harder to predict. We also include some visualization examples of where the attention module obtains more information. These examples provide us with a better understanding of the importance of the character and word features in our results.

4.3.1 What emoji is difficult to predict?

Table 5 shows a list of the top and bottom 10 emojis based on the prediction accuracy. We investigated what emojis are difficult to predict, and we found interesting patterns. As expected, the emojis that are easier to predict describe specific objects without multiple meanings (such as, 🍷 and 🍷) or topics (e.g., 🍷 and 🍷). These emojis, as suggested in [BRS16], could easily be replaced by a word, such as 🍷 by key), or are used when specific words occur in a text message including Christmas for 🍷 and 🍷). In both datasets, subjective emojis including 🍷 and 🍷 obtained lowest accuracy values. These subjective emojis describe emotional information, and they can be interpreted differently among different users and based on the surrounding context. Hence, these emojis do not seem to

have a specific meaning and become difficult to model.

4.3.2 Feature and Word Attention

We previously described the two types of attention explored. The *Feature Attention* approach gives more importance to either the character or word representation of a word. The *Word Attention* approach increases the importance of more discriminative words, for example the word “key” to predict the emoji 🍷.

Figure 5 visualizes the weights of each of these two attention modules using three example messages. For each of them, we list the gold label (“G”) and the predicted labels (“P”), along with their prediction probability. i.e. the output of the softmax layer. The internal weights of the two attention modules are visualized using text highlights. Darker color indicates more attention over word (α_{it} from Formula 2 of each word in the message). In second line of each message the red highlight shows the weights of the feature attention (α of Formula 1). Bold text formatting indicate the out of vocabulary words.

Based on the three examples, and some additional that we manually evaluated, we verified how these two attention approaches work. The *Word Attention* module (blue highlight) give us insights on the recognition of emojis. In the first example the most important word is “cat” and the predictions are indeed about cats, apart from the fifth predicted emoji 🍷. This emoji is triggered (probably) because of the presence of the token “135” as the word attention module also focuses on this token. In the second example, the attention goes to the word “pregnant”, but in this case this word misleads the network that incorrectly predicts baby emojis. However, the correct emoji is then predicted as fourth option. In the last example, the network correctly classifies the emoji 🍷, based on the hashtag “#unicorn”.

Regarding the *Feature Attention* over the word or character representation of each token in a message, we observed that the character representation seems to gain importance on long and less frequent tokens, namely numbers, hashtags, and as expected, out of

vocabulary words (“135” and “#austinblueroos”).

5 Conclusion

In this paper, we explored emoji prediction in two social media platforms, Twitter and Snapchat. We extended the emoji prediction task to a large number of emojis and showed that the prediction performance drastically drops between 50 and 100 emojis, while the addition of more emojis keeps the accuracy of the model somehow constant (even if it has to predict more emojis). We attribute these results to the specificity of the less-used emojis. We also proposed a novel task that predicts broader classes of emojis, grouping emojis in automatic clusters or predefined categories, as defined by the Unicode consortium. These new tasks allow us to better evaluate the predictions of the model, since plain emoji prediction may be ambiguous. We also carried out an extensive qualitative analysis in order to understand the importance of the character encoding of words in noisy social media text, the number of training examples, and the difficulties in modeling specific emojis.

Finally, we proposed a multi-task approach to predict emojis and emoji group affiliation at the same time. We showed that the model obtains significant improvements in the Twitter dataset, while more investigation is needed for the Snapchat dataset.

Acknowledgments

This work was done when Francesco B. interned at Snap Inc. Francesco B. acknowledge support also from the TUNER project (TIN2015-65308-C5-5-R, MINECO/FEDER, UE) and the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502).

References

- [BBRS18] Francesco Barbieri, Miguel Ballesteros, Francesco Ronzano, and Horacio Saggion. Multimodal emoji prediction. In *Proceedings of NAACL: Short Papers*, New Orleans, US, 2018. Association for Computational Linguistics.
- [BBS17] Francesco Barbieri, Miguel Ballesteros, and Horacio Saggion. Are emojis predictable? In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 105–111, Valencia, Spain, April 2017. Association for Computational Linguistics.
- [BCC18] Francesco Barbieri and Jose Camacho-Collados. How Gender and Skin Tone Modifiers Affect Emoji Semantics in Twitter. In *Proceedings of the 7th Joint Conference on Lexical and Computational Semantics (*SEM 2018)*, New Orleans, LA, United States, 2018.
- [Ben12] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *ICML Workshop*, 2012.
- [BKRS16] Francesco Barbieri, German Kruszewski, Francesco Ronzano, and Horacio Saggion. How Cosmopolitan Are Emojis? Exploring Emojis Usage and Meaning over Different Languages with Distributional Semantics. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 531–535, Amsterdam, Netherlands, October 2016. ACM.
- [BRS16] Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. What does this emoji mean? a vector space skip-gram model for t.emojis. In *LREC*, 2016.
- [Car] R. Caruana. Multitask learning: A knowledge-based source of i.b. In *ICML’93*.
- [Car97] Rich Caruana. Multitask learning. *Mach. Learn.*, 28(1):41–75, July 1997.
- [CMS15] Spencer Cappallo, Thomas Mensink, and Cees GM Snoek. Image2emoji: Zero-shot emoji prediction for visual media. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1311–1314. ACM, 2015.
- [CSG⁺18] Spencer Cappallo, Stacey Svetlichnaya, Pierre Garrigues, Thomas Mensink, and Cees GM Snoek. The new modality: Emoji challenges in prediction, anticipation, and retrieval. *arXiv preprint arXiv:1801.10253*, 2018.
- [CW08] R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*. ACM, 2008.
- [ERA⁺16] Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bosnjak, and Sebastian Riedel. emoji2vec: Learning emoji representations from their description. In *Proceedings of The Fourth International Workshop on Natural Language Processing for Social Media*, pages 48–54,

- Austin, TX, USA, November 2016. Association for Computational Linguistics.
- [FMS⁺17] Bjarke Felbo, Alan Mislove, Anders Søgaard, Iyad Rahwan, and Sune Lehmann. Using millions of emoji occurrences to learn any-domain represent. for detecting sentiment, emotion and sarcasm. In *EMNLP*, 2017.
- [GS05] Alex Graves and Juergen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18, 2005.
- [HGS⁺17] Tianran Hu, Han Guo, Hao Sun, Thuyvy Thi Nguyen, and Jiebo Luo. Spice up Your Chat: The Intentions and Sentiment Effects of Using Emoji. *Proc. of ICWSM 2017*, 2017.
- [KK17] Mayu Kimura and Marie Katsurai. Automatic construction of an emoji sentiment lexicon. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, pages 1033–1036. ACM, 2017.
- [LLM⁺15] W. Ling, T. Luís, L. Marujo, R.F. Astudillo, S. Amir, C. Dyer, A.W. Black, and I. Trancoso. Finding function in form: Compositional character models for open vocabulary word representation. *EMNLP*, 2015.
- [MLS13] Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013.
- [NSSM15] Petra Kralj Novak, Jasmina Smailović, Borut Sluban, and Igor Mozetič. Sentiment of emojis. *PloS one*, 10(12):e0144296, 2015.
- [RCP16] Marek Rei, Gamal KO Crichton, and Sampo Pyysalo. Attending to characters in neural sequence labeling models. In *Coling*, 2016.
- [RPG⁺18] David Rodrigues, Marília Prada, Rui Gaspar, Margarida V Garrido, and Diniz Lopes. Lisbon emoji and emoticon database (leed): Norms for emoji and emoticons in seven evaluative dimensions. *Behavior research methods*, pages 392–405, 2018.
- [WBSD17a] Sanjaya Wijeratne, Lakshika Balasuriya, Amit Sheth, and Derek Doran. Emojinet: An open service and api for emoji sense discovery. *International AAAI Conference on Web and Social Media (ICWSM 2017)*. Montreal, Canada, 2017.
- [WBSD17b] Sanjaya Wijeratne, Lakshika Balasuriya, Amit Sheth, and Derek Doran. A semantics-based measure of emoji similarity. *Web Intelligence*, 2017.
- [YYD⁺16] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016.