

Emojis Dataset

Yann MENTHA,
DLab, EPFL, Switzerland
January 14, 2021

Abstract—As emojis became a central part of democratized digital communication in the last decades, being able to accurately represent them in an appropriate semantic space becomes a crucial aspect of Natural language processing when it comes to extract the emotion and meaning of a sentence presenting emojis. Such a representation has to reflect the commonly accepted, potentially multiple, meanings of an emoji and should be able to embed the emojis in the same space as the words themselves.

In this report, we present an overview of the creation of the Emojis dataset, which consists of human labeled single-word descriptions of the 1325 most common emojis (30 annotations per emoji in average). We also proceed to an in-depth presentation of the various tools and methods developed in order to collect the data.

I. INTRODUCTION

Emojis are a great manner to provide valuable insights on the emotions and contexts of a sentence extracted from conversation data or social network posts. Nevertheless, in many cases the emoji chosen by the writer of the message does not share the semantic of the rest of the sentence, but rather completes it. Indeed, in the sentence "No way, did you go to the park? 😂" no word relates to an euphoric feeling, as the same sentence "No way, did you go to the park? 😡" can carry a totally different message. Therefore, using the words surrounding an arbitrary emoji in a similar manner as in a skip-gram model [1] does not seem like the most promising method to model them. In the Emoji2vec paper [2], researchers tackled this issue by using a few expert-labelled sources of descriptions for emojis. However this approach might miss some commonly-accepted meanings, as a same emoji can be used in many contexts and have different meanings each time, besides the fact that the usage of emojis and their respective meaning might vary over time.

Hence, a dataset containing human-labeled descriptions of the most common emojis could allow reserchers to train NLP models on data that better fit the meaning the general population gives to emojis. As no such dataset exists in the literature at the time of writing yet, we aim at filling this gap.

II. MODELS AND METHODS

The approach we followed consists of the following parts:

- Selection of emojis:** data-driven manner of selecting which emojis to keep in the dataset
- Quantity needs:** data-driven manner of choosing the number of answers per emoji
- Data Collection:** techniques used to create the forms used to gather data and monitor workers in real time
- Post Processing:** data cleaning and refactoring
- Validation:** t-sne visualization of a word2vec/bert embedding created on-top of the Emojis dataset

A. Selection of Emojis

Although more than 3859 UTF-8 emojis exist at the time of writing of this paper (emoji pypi package, v. 0.6.0), it turns out that only a slight portion of them cover the major part of social network use (Fig. 1), in the same way that we tend to only use a small portion of the english vocabulary on an everyday basis. Therefore, we decided to focus more resources on useful emojis rather than producing a balanced dataset which would miss data for critical emojis and give too much weight to marginal ones. The covered ratio of emojis use w.r.t emojis sorted by their frequency is shown on Fig. 2 .

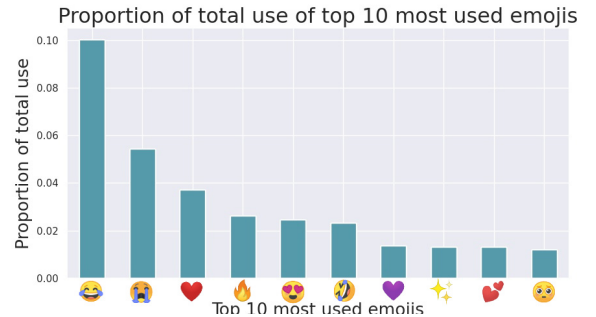


Figure 1: **Top 10 emojis** : ratio of total use for the top 10 most used emojis on 1Gb of tweeter data. These 10 emojis account for 31.72% of the total use. Data extracted from 1Gb of random english (u.s) tweets in the year 2018

As tweeter data does not represent the only context in which our dataset should be useful, we performed the selection of emojis by taking into account frequency and practicality. (Removal of hours, flags, letters, numbers emojis.) Furthermore, we aimed at keeping clear from gendered and skin-toned emojis by only keeping the neutral corresponding version (yellow for the skin) and mapping back these emojis back to their neutral version.

Our final emojis selection covers 94.7% of the total emojis use of our tweeter data, and represents 34.34% of all emojis.

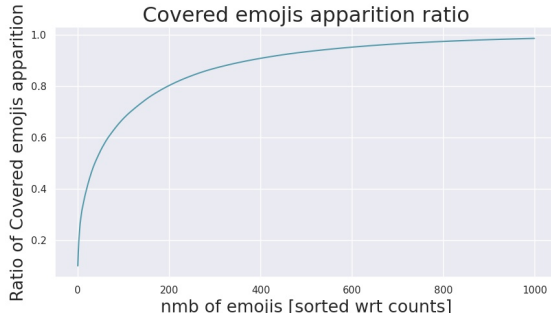


Figure 2: **Covered emojis apparition ratio** : the vast majority of emojis use is covered by a small subset of emojis. The top 600 most used emojis covers $> 95\%$ of total emojis use.

B. Quantity needs

Depending on their nature, emojis can require more or less words to describe them accurately. For instance, an emoji as simple as 🔥 could intuitively be described by "fire" or "burn" whereas 😞 would present a broader lexical field, including words such as "confused" "skeptical" or "thinking".

As such, we had to define a data-driven limit on the number of annotations to collect per emoji: gathering a large amount of annotations for 🔥 could turn out being pointless just as gathering too little data for the 😞 could not represent its true distribution. (😞 was described by 13 distinct words on 30 annotations whereas 🔥 vocabulary only consists of "fire" and "hot")

Distribution shift

The point is to collect data as long as some new information is added to the existing distribution, or, in other words, as long as we observe a shift of the distribution in question. We therefore chose as a metric the convergence of words distribution for every emoji.

First, a pilot dataset of 12 emojis and 160 annotations per emoji was gathered. Every such annotations were

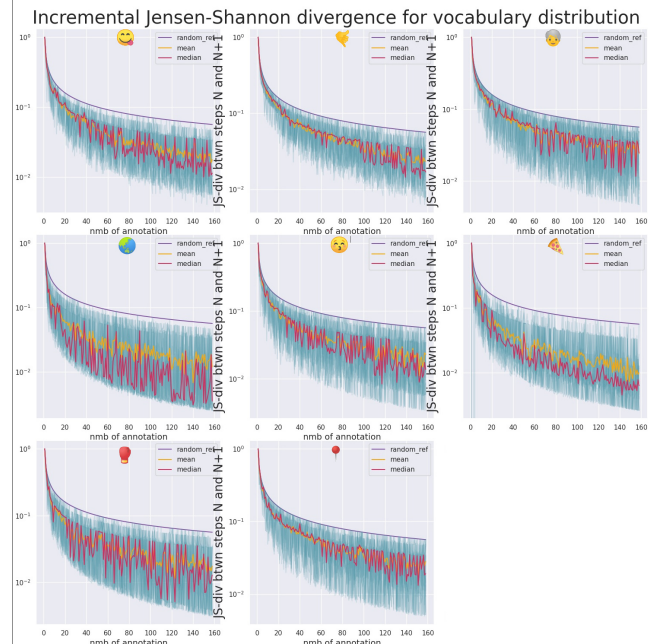


Figure 3: **Incremental JS distance**: the Jensen-Shannon (JS) divergence of the distribution before and after the addition of an annotation to a specific emoji vocabulary tends to decrease for two reasons. First, as the number of annotations increases, the newly added sample represent a fraction decreasing as $\frac{1}{n}$. As such, its effect on the distribution shift will be less and less impactful. The second reason is the desired one : we observe a stabilization of the distribution as the most used words are covered by the gathered vocabulary. An annotation using a word already seen many times will therefore have less impact than a brand new word. The purple curve represents a synthetic word distribution that add new at each step and serves as a reference.

assumed to be statistically independant given that the workers ids are ensured to be all distinct. The fact that every such annotation were made at the same moment in time was not taken into account in this assumption. Once this was done, we computed trajectories for every emoji: such a trajectory consists of a random ordering of the words used to describe a given emoji in the pilot dataset. We computed the Jensen-shannon (JS) divergence between $f_n(w)$ and $f_{n+1}(w)$ with $f_n(w)$ the distribution of words including all words up to $n \in \mathbb{N}$ in the trajectory. The experience was repeated several times in order to reduce the variance due to the stochastic nature of such trajectories. Figure Fig. 3 shows the Jensen-Shannon divergence for 20 of these trajectories per emoji for a set of emojis selected to represent several levels of complexity.

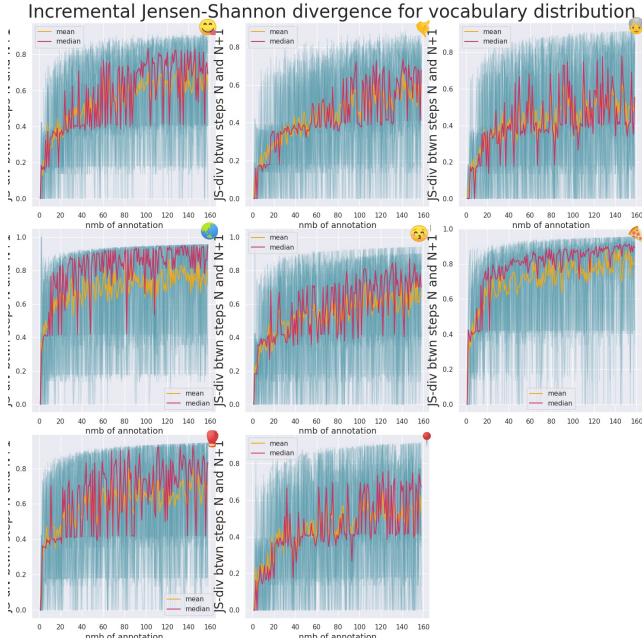


Figure 4: **Normalized Incremental JS distance**: similar plot as in Fig. 3, where each curve was normalized w.r.t the artificial curve in order to get rid of the converging behavior due to the scaling of the distribution.

The shift of distribution induced by the addition of a single new word gets significantly smaller as the number of samples n present in that distribution increases. Indeed, the JS convergence of a trajectory presenting a new word at every step still converges, as can be seen on the purple line of figure Fig. 3, where such a trajectory was artificially computed. The divergence evolution of every emoji was therefore normalized w.r.t to this artificial trajectory in order to obtain relevant insights concerning the convergence of distribution shifts as $t_{norm}(n) = \frac{t(n)-r(n)}{r(n)}$ with $t(n)$ a trajectory, $t_{norm}(n)$ its normalized version and $r(n)$ the artificial reference. Fig. 4 highlights this behaviour.

Despite of a significant noise level in the signal due to the limited size of the pilot dataset, it seems like several emojis converge after 120 annotations per emoji. Due to practical constraints we had to reduce this number to 30 annotations per emoji in the final dataset, which might not represent accurately the true distribution of certain complex emojis. 🍕 was described with 2 words, whereas 🍌 had a vocabulary size of 13, in consistence with the obtained results.

Number of words per annotation

A crucial point concerned the number of words a worker could give for one emoji: many annotations can be necessary to describe complex emojis, but useless for simpler ones. We therefore tested two frameworks: the first allowing 3 words per emojis, the second allowing 1 word only. As similar results quality was obtained in both cases, the second framework was preferred for its simplicity.

C. Data Collection

In order to collect human-labeled data, we chose the Amazon MTurk crowdsourcing marketplace (Mturk) for its convenience and the presence of a python library and api (boto3) allowing automation of the tasks creation and control. The annotations had to be collected through forms presenting the following properties:

- On-the-fly answer format checking
- Confirmation code checking
- Cross-browser compatible emojis formats.
- Automatic form creation from emojis list

As Mturk did not present such native features, we combined it to Google Forms (Gform) using Appscript to automate the forms creation. Every MTurk task was then linked to a corresponding Gform.

Mturk2Gform

In order to ensure a minimum diversity in the source of the data, we had to limit the number of forms one worker could answer. Fig. 5 Neither Mturk nor Gform provided such control: we therefore implemented it along with all features cited above in a new python package: Mturk2gform. The library allows notably for synchronization of the Mturk framework with Gforms, automation of Mturk HITs creation/handling, and control of the number of forms answered per worker. (cf Fig. 6). Even though control of data quality could have been carried out on-the-fly using Mturk2gform, all workers inputs were accepted fpostprocessingor the sake of Mturk requester reputation: bad feedbacks from workers make it indeed less likely for future forms to get answered.

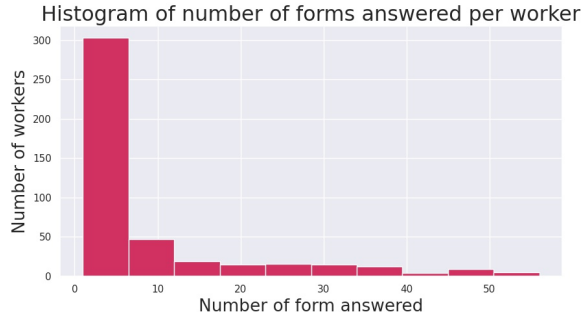


Figure 5: **Number of forms answered per worker:** The high number of workers answering a small amount of forms ensure some minimal diversity in the data. Mturk2gform was used to set a maximum limit at 50.

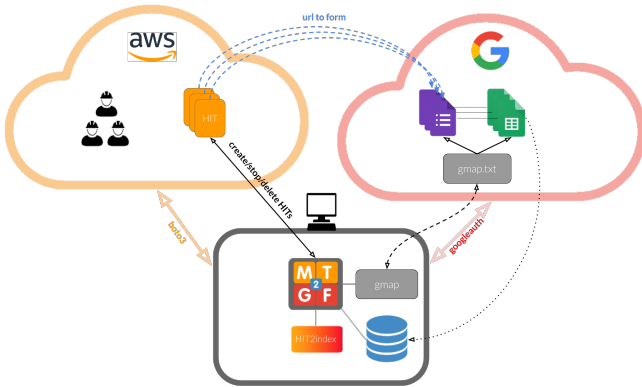


Figure 6: **Mturk2gform:** Mturk2gform (mt2gf) allows one to synchronize the mturk framework using google forms. A one-to-one mapping between Mturk HITs and google forms indexes is maintained (hit2indx). Once the forms are created, mt2gf creates Mturk hits allowing workers to access the gforms through a link. During the time when workers answer the forms, mt2gf regularly download a local version of the results for quality checks/fraudulent workers tagging.

Age, gender and mothertongue informations were collected from the workers as well.

The overall characteristics of the dataset can be found in Table I and Fig. 7

Table I: Emojis Dataset characteristics.

Total nmb of emojis	1325
Nmb of forms	118
Nmb of emojis per form	10
Max nmb of forms per worker	56

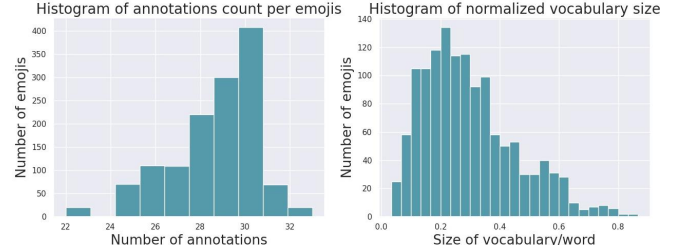


Figure 7: **Annotations and vocabulary size per emoji:** (left) Due to removal of invalid inputs (cf Sec. II-D) some emojis ended up with less than the expected 30 annotations. (right) Plot of the normalized vocabulary: emojis having received less annotations are more likely to have a smaller vocabulary size. We therefore display the vocabulary size per annotation, showing a mean of 0.3 words/annotation, corresponding to a vocabulary size of approx. 10 for 30 annotations.

Display format

The pilote datasets revealed that UTF-8 emojis visualization could be problematic depending on the browser type and version: some browsers would display a default rectangle instead of the actual emoji, making it impossible for workers to describe it in a meaningful manner. We therefore decided to display emojis as images instead of using the plain text utf-8 format. As no dataset mapping emojis to their corresponding image existed, we created one by automatically capturing a screenshot of the sequentially displayed emojis in a jupyter notebook.

D. Postprocessing

A post-processing pipeline was implemented to detect malformed data. Each of the 3 steps presented in this subsection can be optionally turned on or off when generating the usable version of the dataset.

Repeated answers

It was observed that a slight minority of workers give a constant answer to every emojis in a form regardless of their aspect. We therefore discarded workers whose vocabulary size was inferior to a given ratio of the maximum number of answers for the form.

Honeypots

One obvious emoji sampled from the honeypots emojis list was placed in every form: this way workers providing answers not within a Levenshtein distance of 3 of any word in the expected honeypot vocabulary were discarded.

Spelling

As the spelling of annotations could not be checked server-side in the Gforms, we had to carry it out as a

post-processing step. This way, a misspelled word would go through the following steps:

- 1) If the word is used by multiple workers to describe the emoji, it is considered valid.
- 2) If the word is a valid english word (enchant python library used as reference) it is kept as is.
- 3) We compute suggestions for the word in question, that is, valid words that are within a maximum Levenshtein distance of 2. If some of these suggestions have been used by other workers, we keep the suggestion that has been used the most frequently.
- 4) If the word can be split into several valid words, we keep the splitted version separated by a ' ' (ex: "uparrow" becomes "up arrow")
- 5) If the word is still considered as invalid, we reuse the suggestions of step 3 and select the one which obtain the highest score in the logits computed by an uncased BERT for masked ML model, using as context all the valid words gathered for the emoji in question
- 6) If no suggestion was available at step 5, we keep the word as is.

```
Modified: riceball --> rice ball (disassembled2)
Modified: ricebowl --> rice bowl (disassembled2)
Modified: ufo --> ufos (corrected)
Modified: omlette --> omelette (corrected)
Modified: omlett --> molest (corrected)
Modified: frenchfries --> french fries (disassembled1)
Modified: vegetable --> vegetable (cross_suggested)
Modified: raddish --> radish (cross_suggested)
Modified: sweetpotato --> sweet potato (disassembled1)
Modified: cakepops --> cake pops (disassembled2)
Modified: kebob --> kabob (cross_suggested)
Modified: popcakes --> pop cakes (disassembled2)
Modified: smores --> s mores (disassembled2)
Modified: kabbobs --> kabobs (corrected)
Modified: smores --> s mores (disassembled2)
Modified: smore --> s more (disassembled2)
Modified: bento --> bent o (disassembled2)
Modified: stricker --> tricker (corrected)
Modified: bottlecap --> bottle cap (disassembled1)
Modified: softserve --> soft serve (disassembled1)
Modified: creamee --> creamer (corrected)
Nof found: icecream
```

Figure 8: **Word correction:** sample extracted from the log of the word correction algorithm

The results shown in the Sec. III Result section were obtained by discarding the workers whose vocabulary size was inferior to 0.8 of the maximum size. Honeypots and spelling corrections were used as well.

E. Validation

As the purpose of this dataset consists in training NLP models, we used one of the multiple applications that could be run on top of it as a quality check. For

comparison purpose, we decided to reproduce the t-sne visualization of embeddings obtained in the emoji2vec paper [2] with both the original data and ours. This dimension reduction is performed on embeddings created using a word2vec model. Ideally, the word classification test (determining whether a given word is associated with an emoji on a dataset with negative sampling) present in this paper should have been ran as well: unfortunately, the experiment turned out to be challenging to reproduce. We therefore kept ourself to the t-sne embeddings. The same experiment was performed using the last attention layer of an english-trained BERT model as embedding in order to check if the quality difference between our dataset and the emoji2vec baseline would be reproducible.

Parameters

The parameters for the word2vec pipeline were kept identical as in the word2vec paper. (40 epochs with a 1-1 negative-positive sampling, 300 input dimension and 300 output dimension). All t-sne plots were performed with 5000 iterations with a perplexity of 30 (sklearn manifold library).

III. RESULTS

A sample of the final dataset is display in Table II. Table III and Table IV show subset of results gathered for emojis presenting respectively most and least diverse vocabulary. Table V shows a subset of the demographic information gathered along with the dataset.

The T-sne embeddings reproduced using the original emoji2vec data can be seen on Fig. 11 and Fig. 11 (word2vec and bert methods respectively): the same is done for our own dataset on Fig. 10 and Fig. 12.

Table II: **Random sample**: example random sample of the emoji dataset















WorkerID	FormId	Duration	emoji _i <i>index</i>	emoji	word
A2CK0OXMPOR9LE	41	52.0	725		graph
A3774HPOUKYTX7	52	458.0	257		rose
A2D2JX8R0QU9G4	51	454.0	642		worker
A3VL1CBZ3BGQK7	28	153.0	83	?	medical
A39MKVROUZ1UWR	95	460.0	208		morning
AB66CTVQ90RCV	92	181.0	1023		walking
A2KX8SX5M47GPX	123	98.0	1255		vessel
A3OP9SCMH0KPJL	20	455.0	223		moon
AOMFEAWQHU3D8	92	793.0	957		upset
A1S8KMGKJACGVO	36	318.0	638		woman
AF0M3066S5UF1	44	89.0	755		mail
A34NDXO56MBC3D	87	554.0	1210		minuscule
AOMFEAWQHU3D8	45	179.0	1235		knee
A33DVD9ZMS0XXN	28	90.0	64		aries
AMA18W8F60Y2J	65	616.0	1320		chair

Table III: **Varied emoji**: Example of words sampled from an emoji presenting a rich vocabulary











WorkerID	FormId	Duration	emoji _i <i>index</i>	emoji	word
A337Y4X67PY4QI	83	117.0	186		key
ATP2BJPDK4K2B	83	233.0	186		box
AJRY9ALX8069Y	83	84.0	186		rewind
A2T007HZK66WM	83	168.0	186		cc
A2MOKIEQZ0OF2M	83	NaN	186		clip

Table IV: **Constant emoji**: Example of words sampled from an emoji presenting a poor vocabulary

WorkerID	FormId	Duration	emoji _i <i>index</i>	emoji	word
A1N1ULK71RHVMM	42	432.0	492		tiger
A2T1LNI80EPOQR	42	466.0	492		tiger
A1J1ZM062PH3FN	42	277.0	492		tiger
A1DD23J1WBGQUU	42	363.0	492		tiger
AOMFEAWQHU3D8	42	279.0	492		tiger

TSNE embedding of e2v using w2v

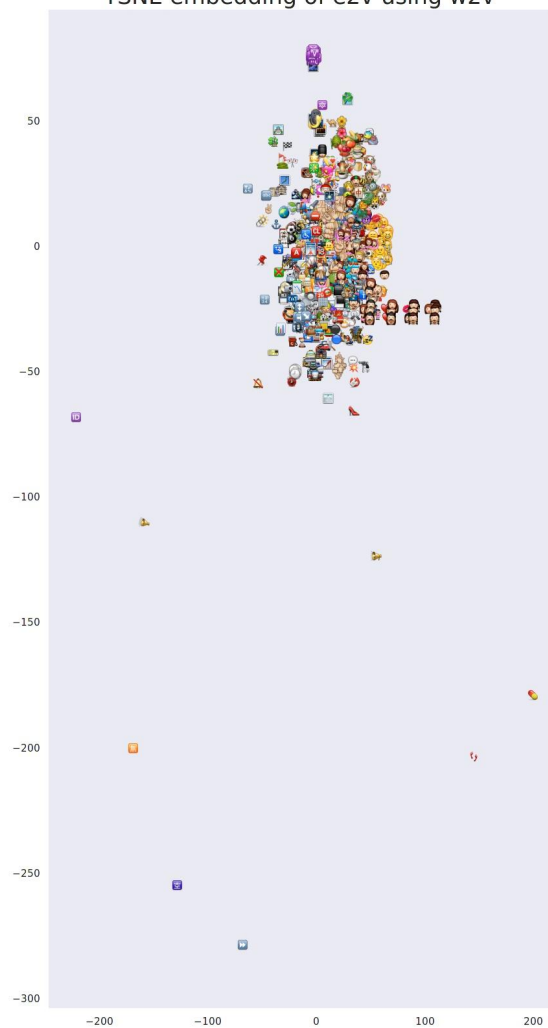


Figure 9

TSNE embedding of em_dataset using w2v

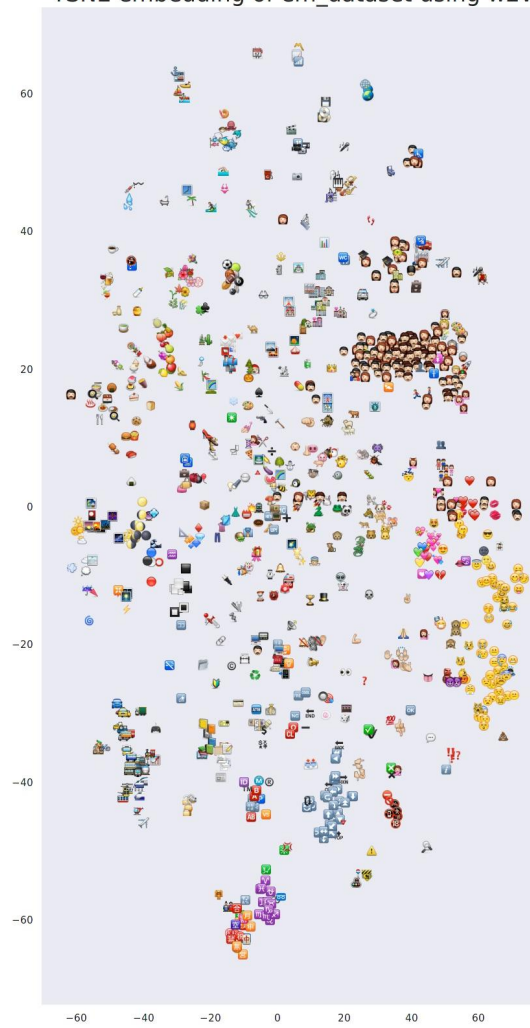


Figure 10

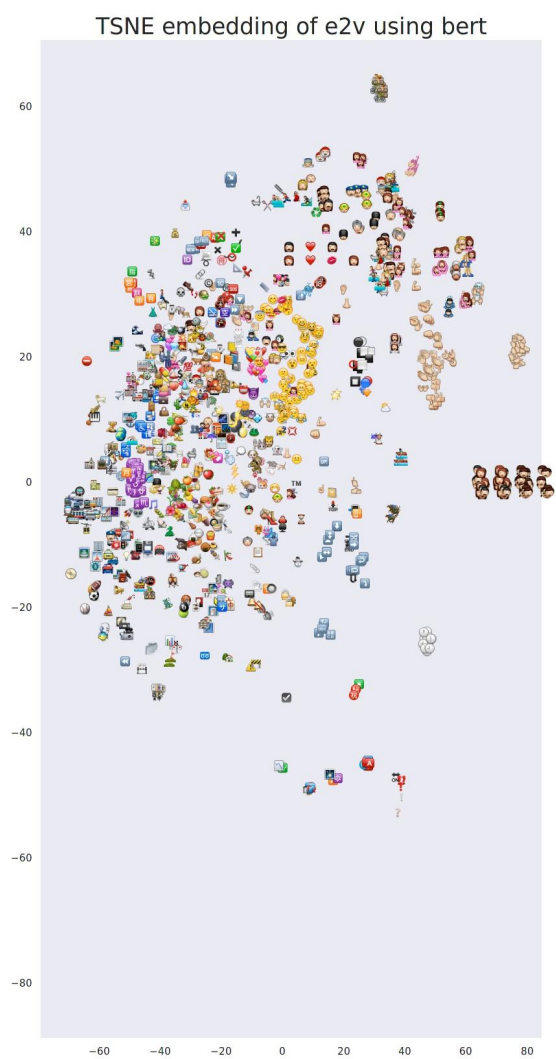


Figure 11

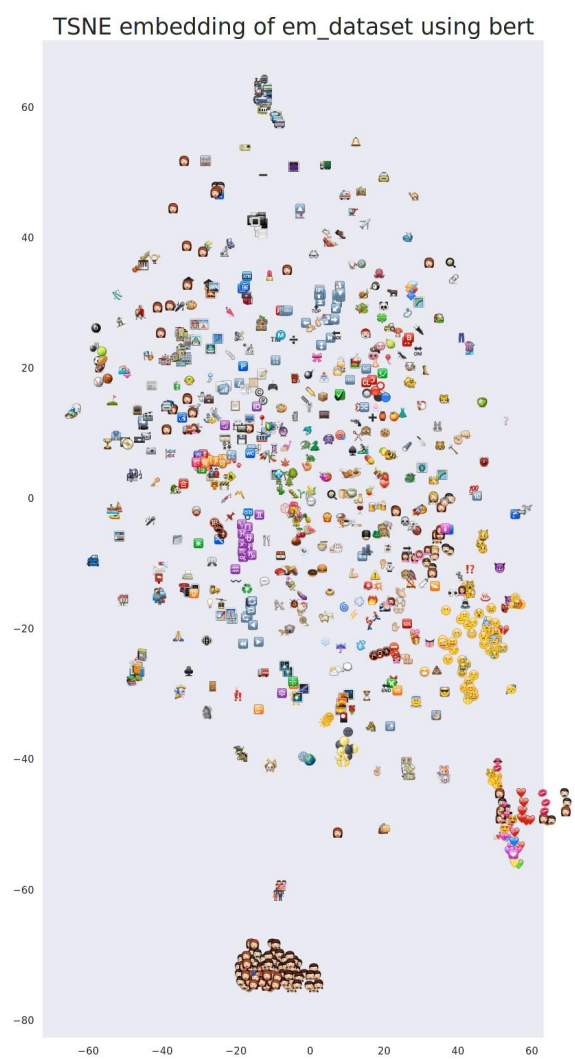


Figure 12

Table V: **Workers Info**: Subset sampled from the demographic workers information table generated with the dataset

WorkerID	Age	Gender	Mothertongue
ALML8V38FDV0	55.0	Female	english
A31UXXZVI3U4E2	31.0	Male	english
AQ53YJDPDDLZ	40.0	Male	english
A143XRCI1YXAFE	28.0	Male	english
A1KR49KPV5J9BM	36.0	Male	english

IV. DISCUSSION

Although some emojis are not properly rendered due to the font used (Apple Color Emoji.ttf, many emojis mapped to only "man" or "woman"), it seems like our dataset led to slightly better embeddings than the emoji2vec baseline, which converged to a compressed cluster. However some inconsistencies persist: zodiac signs are spread across the image, food cluster is not very centered etc. Bert embeddings lead to slightly more defined clusters with this regard.

Overall the produced dataset looks usable for many NLP applications: if more data was to be needed in the future, the mturk2gform framework would facilitate the gathering of additional annotations.

REFERENCES

- [1] Kenneth Ward Church. Word2vec. *Natural Language Engineering*, 23(1):155–162, 2017.
- [2] Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. emoji2vec: Learning emoji representations from their description, 2016.