A Quantitative Study On How Emojis' Meanings Shift From West To East: A CIS545 Report

Mingyang Li*

Abstract

Interpretations to emojis vary from culture to culture. For example, the folded-hand emoji often reminds western audiences of preyers, but would be no more than a "thank you" to native Japanese speakers. Such discrepancies have been long observed but hardly quantitatively measured. In this paper, I present a methodology of extracting the emotional components associated with each emoji from a large corpus, and – with selected emojis – demonstrate how to interpret the difference across corpora.

1 Introduction

Same emojis can resonate differently to people of different culture[4]. It is an interesting topic for linguists, and perhaps an important aspect to consider for creative designers when employing emojis into international products. However, related quantitative research often focuses on usage statistics with a country-level granularity, such as the SwiftKey Emoji Report on April 2015.[3] Blanks are still to be filled when it comes to quantifying the emotions associated within an emoji. In this paper, I present a pipeline that extracts the emotion components associated with any emoji and demonstrates with some emojis how to discover difference in interpretations towards each emoji across cultures.

2 Data Preprocessing

2.1 Selecting Dataset

Data are fetched from two microblog services in U.S.A. and China, representatively: *Twitter* and *Sina Weibo*. Two considerations motivated such choices:

- **Microblog services are chosen as data source** because their contents are often colloquial and opinionated. Compared to news articles and book scans, microblog corpora are expected to (1) employ more emojis and (2) express more emotions.
- ***Twitter* and *Weibo* are chosen** due to their relatively isolated user base.
  - Demographically, *Weibo* is almost exclusively used by Mainland China citizens, to whom *Twitter* is unaccessible due to governmental censorship.
  - Linguistically, I filter *Twitter* for English tweets and *Weibo* for Chinese posts. A total of 0.35 billion documents are selected.

2.2 Cleaning Data

Data cleaning is mainly focused on the *Sina Weibo* side. The procedure involves:

1. **Splitting each post into separate "repost portions."** This is because *Weibo* **prepends** retweeter's comment to the retweeted content, rather than storing them separately like *Twitter* does.
2. **Removing direct retweets** – reposts with no additional comments.

*myli@seas.upenn.edu.

3. **Filtering out portions written in languages other than Chinese**. Languages are identified with *langid.py*, a Python package.

4. **Segmenting Chinese sentences.** This is because Chinese sentences are not naturally segmented with whitespaces.[10] Among many segmenters available for Chinese corpora, *Jieba* is chosen for its ability to discover new words and slangs, which is particularly important for a highly colloquial corpus like *Sina Weibo*.

For both corpora, the following types of tokens are conformed together to reduce vocabulary size:

- **"At" mentions.** For example, "Had a great time with my friend @username tonight" is conformed to "Had a great time with my friend {AT} tonight".
- **Hashtags.** "We went to watch #TheAvengers at 6" is conformed to "We went to watch {TP} at 6". "TP" stands for "topic", as *Sina Weibo* calls it.
- **URLs.** "Check out this: http://url.cc/YhGs6PN" is conformed to "Check out this {LK}".
- **Retweet mentions**. "RT @username: Retweet to show support!!!" is conformed to "{RT} Retweet to show support!!!".

The whole preprocessing pipeline is written with PySpark and executed on a Hadoop cluster. The conformation and tokenization are accomplished using regular expressions wrapped as user-defined Spark functions for performance and maintainability concerns.[1] The prepared datasets measure as follows:

- *Twitter*: 187.6 million documents in 11 GB.
- *Weibo*: 342.4 million documents in 50 GB.

3 Implementation and Analysis

1. For *Twitter* and *Weibo* separately, train a *Word2Vec* model with the *gensim* package.
2. For each model, get word vectors of all emojis available on the corresponding microblog platform.
3. Use t-SNE to reduce their dimensions to 2, effectively clustering them according to actual usage in context.
4. Plot emojis using their 2D vectorial representations as coordinates. This helps evaluating the quality of a model.
5. After model is adequately trained – marked by several clearly-separated clusters on the aforementioned plots – for each emoji in a list of common ambiguous emojis, calculate their similarity to each of five typical emojis. The latter group of emojis represent the five basic emotions of humankind.
6. Each ambiguous emoji should now have two 5D vectors assigned to them, derived from the two microblog platforms respectively. Plot each pair of these vectors onto a 5-axis radar plot, with one polygon for each dataset and each axis representing a typical emoji.

3.1 Training *Word2Vec*

Word2Vec is a word embedding model capable of analogy computations such as "King - Man + Woman = Queen".[8]

Implementation Selection. Two implementations of this algorithm, `spark.ml.feature.Word2Vec` (native to the Spark framework) and `gensim.models.word2vec`, are considered.[1, 2] In the Spark edition, only the training data – not the computation itself – is distributed. Such lack of distributed computation implies that the two implementations should consume identical amounts of CPU time. To make it worse, distributing the dataset over a Spark cluster requires additional I/O time, too. Considering that 11+50=61 GB is not a problem for local storage, Spark's *Word2Vec* renders no benefit to our purpose. The *gensim* edition is preferred over the Spark implementation.

Data streaming. A hard disk can easily accommodate 61 GB of data, but not a memory. Fortunately, The *Word2Vec* algorithm supports "online training", where documents are consecutively fed into the trainer program.

[1] It is worth noticing that no lemmatization nor stemming was employed, because word form is important in sentimenta analysis, compared to search engine optimization.

In fact, the creator of *gensim*, Radim Řehůřek, has recommended streamed training in one of his tutorials.[9] This file streaming technique is heavily employed by my project.

Hyperparameters. Due to the relatively long duration of training (~10 hours for Weibo), hyperparameter optimization was not realistic for this project. Therefore, most configurations are kept with their default values. In summary,

- The Continuous Bag-Of-World (CBOW) edition of the *Word2Vec* algorithm is selected.
- The learning rate is initialized at 0.025 and allowed to be dropped to as low as 0.0001.
- 5 epochs are computed over the whole dataset. Another 5 are used to make sure the model had converged.
- Raw word vectors learned consists of 100 dimensions.

3.2 Reducing Dimension With t-SNE

To evaluate the quality of learned model, clustering vectors into 2D will be an intuitive approach. Popular algorithms in dimension reduction includes Principal Component Analysis (PCA) and T-distributed Stochastic Neighbor Embedding (t-SNE). The former attempts to grasp as much as possible global trends of the given dataset, while t-SNE focuses on gathering together locally proximate items.[6] t-SNE not only suits our very purpose better, but also has already been widely adopted by literature.[5]

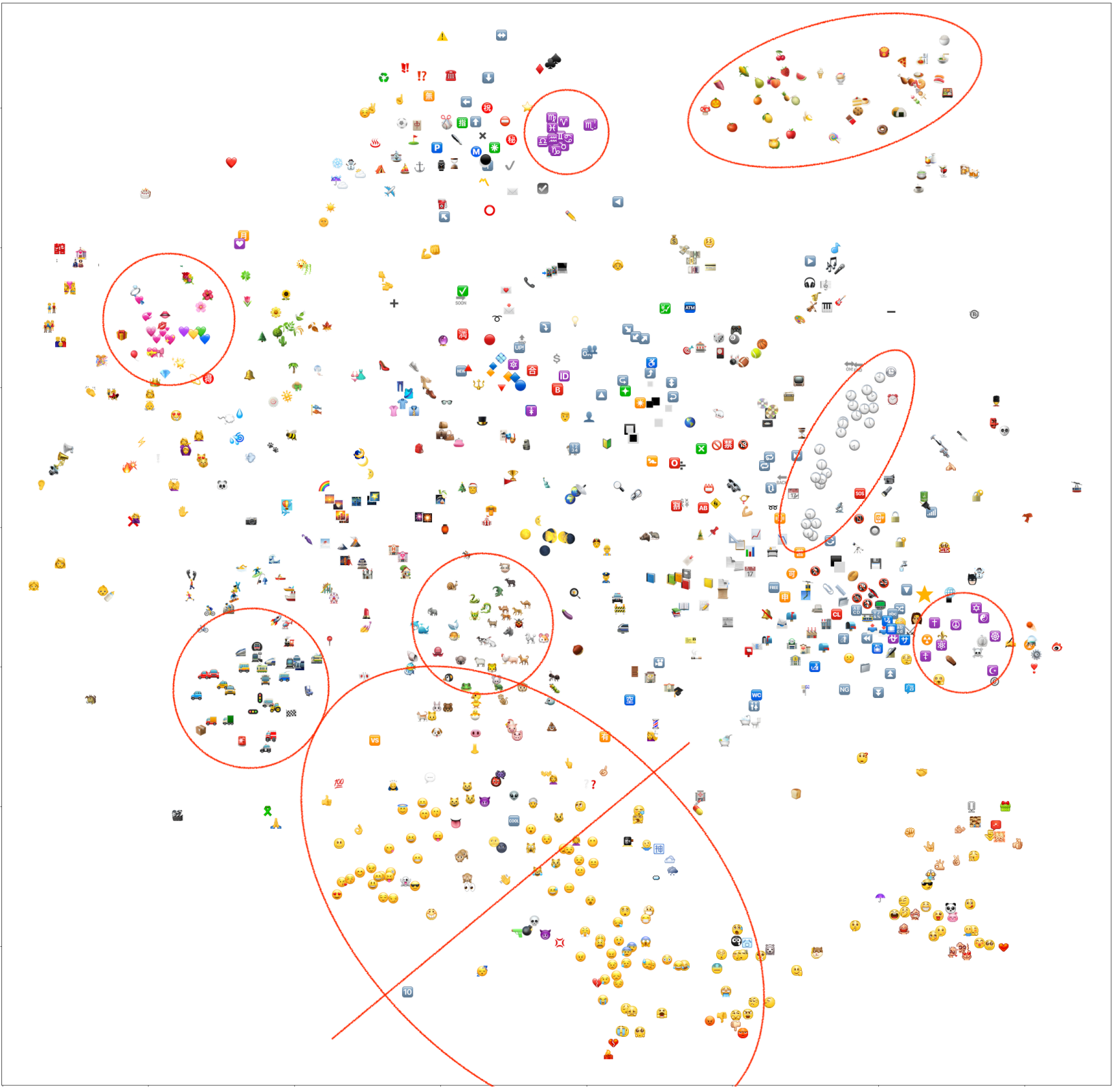

Figure 1: t-SNE embeddings of emojis used on Weibo

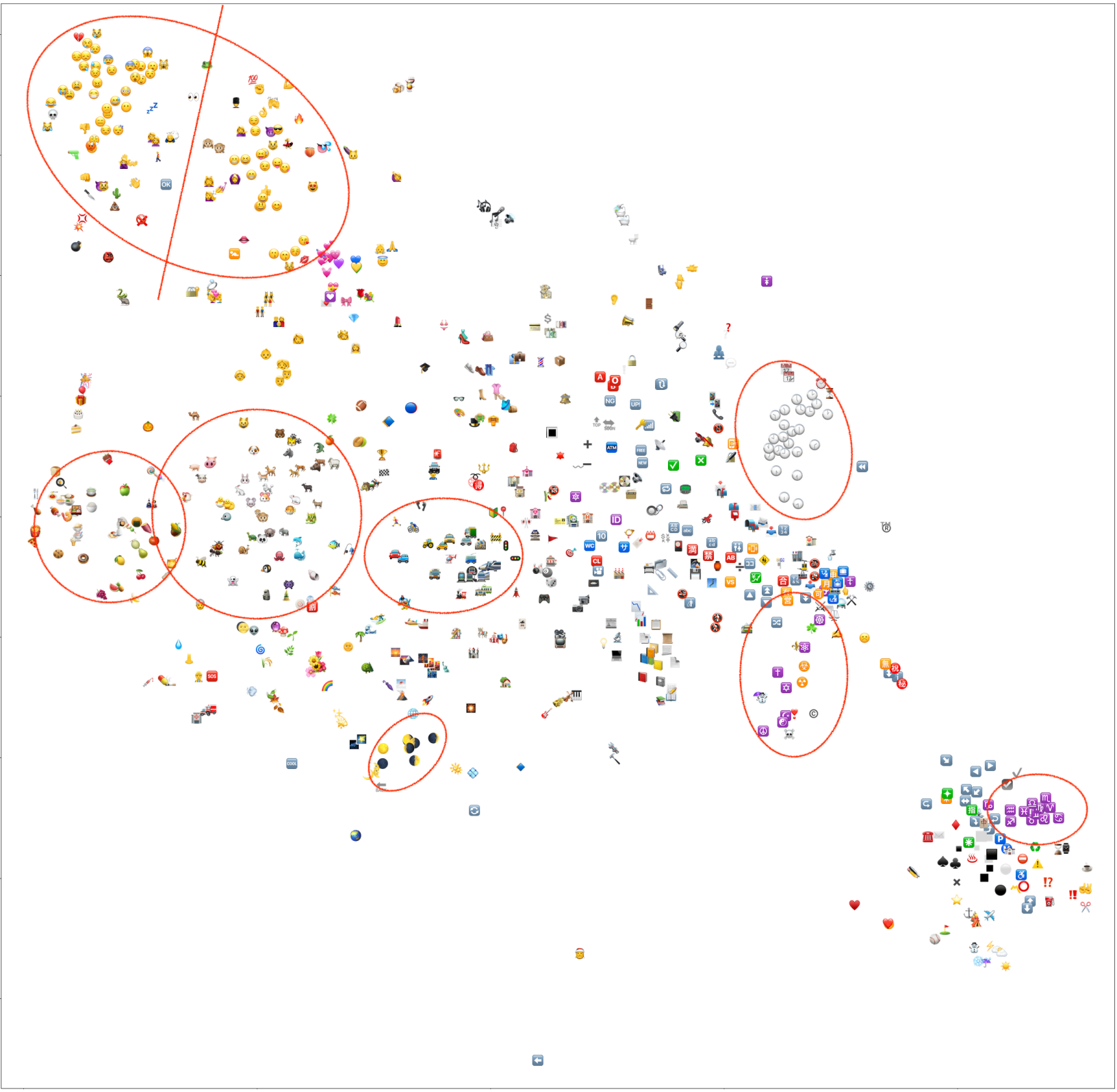

Figure 2: t-SNE embeddings of emojis used on Twitter

Shown here are two scatter plots of emojis corresponding to *Weibo* and *Twitter* corpora, respectively. In both images, emojis alike are clustered nicely together, and their proximity indicates that they are used "interchangeably" across many similar sentences. For example, clock-face emojis are gathered up at the center-right sections of both plots, which may be resulted from the common usage of the sentence structure, "Let's meet at [clock-face emoji

here]". Similarly, the cluster of transportation tools may be a result of many "Let's go there by [transportation emoji]" sentences. These clearly separated clusters here suggests that our *Word2Vec* training has been successful in capturing usages of emojis.

3.3 Projecting Each Emoji Onto Five Basic Emotions Of Humans

3.3.1 Selecting Emojis of Interest

Compared to these non-living objects, we are interested in emojis with facial expressions.

Definition 1. Emojis of interest are emojis with a face.

Before projection, we want to remove shard vectorial components throughout all emoji vectors. Those components include:

1. Component distinguishing emojis from textual tokens. This comes from the fact that all vectorial representations are learnt together.
2. Component distinguishing emojis with faces from those without, due to our selection of emojis of interest.

This can be easily achieved by subtracting the mean of every dimension from each vectors.

3.3.2 Selecting Axis

According to psychologists Paul Ekman and Wallace V. Friesen, all human emotions can be broken into six basic emotions: anger, disgust, fear, happiness, sadness and surprise. Due to lack of expressed disgust in both corpora, we have to settle with the remaining five as our axis.

Definition 2. Basic emotions are anger, fear, happiness, sadness and surprise.

Definition 3. Basic emojis are 😊, 😢, 😱, 😡, 😵.

An experienced emoji user may argue how this seemingly arbitrary selection of emojis can accurately represent the five basic emotions. This concern stems from the fact that the mapping between emotions to emojis is hardly one-to-one:

1. **For an emotion, many emojis can be used to represent it.** For example, one may use any of 😙, 😮, 😵, 😳, 😨 and 🙀 to express "surprise".
2. **For an emoji, many emotions can be inferred from it.** For example, 😨 may mean "surprised-ness", "fearfulness", or a combination of both. In fact, dissecting the emotional components of an emoji is exactly the goal of this paper.

We tackle this problem by orthogonalizing – via Gram-Schmidt method – the basic emoji vectors. In plain English, we **wiggle** these five vectors till they become perpendicular to each other. Here, "**wiggling**" implies that we achieve **orthogonality** at the **lowest** expense of steering each vector away from its original orientation. Key concepts here are "lowest expense" and "orthogonality":

- This **lowest-expense** approach ensures that the **essence** distinguishing basic emojis from each other is captured. This mean that orthogonalized basic vectors represent these fundamental differences, rather than the "basic emojis" themselves.
- On top of that, **orthogonality** guarantees that this **essence** is the **only** component preserved. In other words, orthogonalized vectors should share no common components.

These mutually orthogonal vectors, once normalized, spans a 5D Euclidean space, where cosine similarity can be computed as projections.

3.4 Interpreting Results

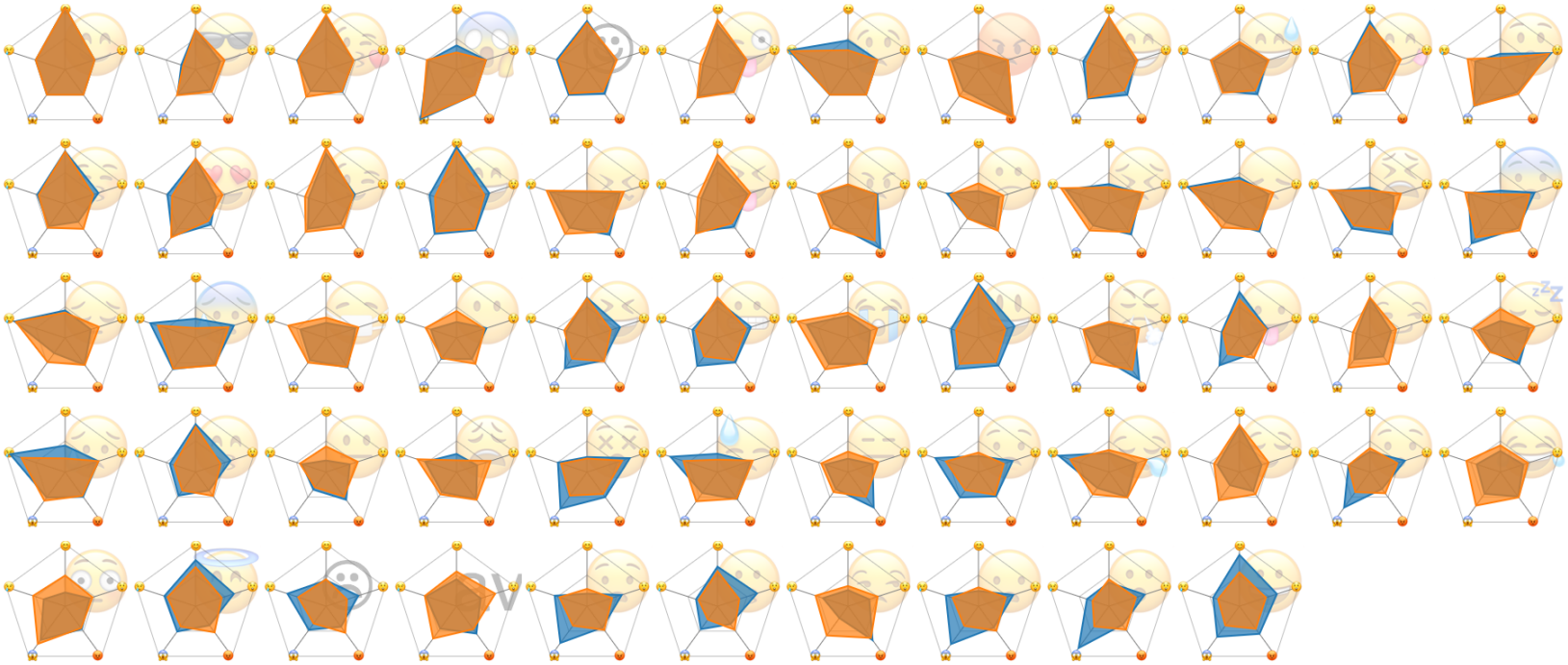

Figure 3: **Projections of 57 emojis with a face onto five emotions.** *D3.js*, a Javascript-based library for data visualization, is used to render radar plots. In the figure, blue polygons represent Twitter, and orange polygons represent Weibo. Emojis of interest are sorted in descending order of polygon similarity. Two levels, shown as grey-lined and unfilled pentagons, mark the 0 and 100% similarities on each radar plot. Notice that the center of each plot indicates a similarity of -100%, rather than 0% as usually seen on radar plots. A below-zero similarity between an emoji and an emotion indicates that this emoji **negates** that emotion.

Reflecting on the fact that disgust is a rarely pronounced emotion, we have to realize that different emotions are expressed at different intensities. In other words, microbloggers may generally restrain themselves from expressing certain emotions, either intentionally or limited by the vagueness of emojis designed to represent that very emotion. Such restriction leads to emojis not stretching to the $\pm 100\%$ boundaries of the corresponding axis. No vertices of the polygons shown in this figure is an evidence of this conserved expression.

A quick fix to this issue is standardizing data over each axis, so that, for each of the five axis, there is at least one radar plot attaining its outermost boundary and another attaining its origin point. This standardization is done over two datasets separately. Keep in mind that we will lose the ability to compare numerical values within each radar plot. For example, the middle pentagonal wireframes will lose their meanings as the 0% baselines. Despite of this downside, this visualization facilitates interpretability across radar plots on corresponding axis. We can choose from the raw projection and standardized projection as our need changes.

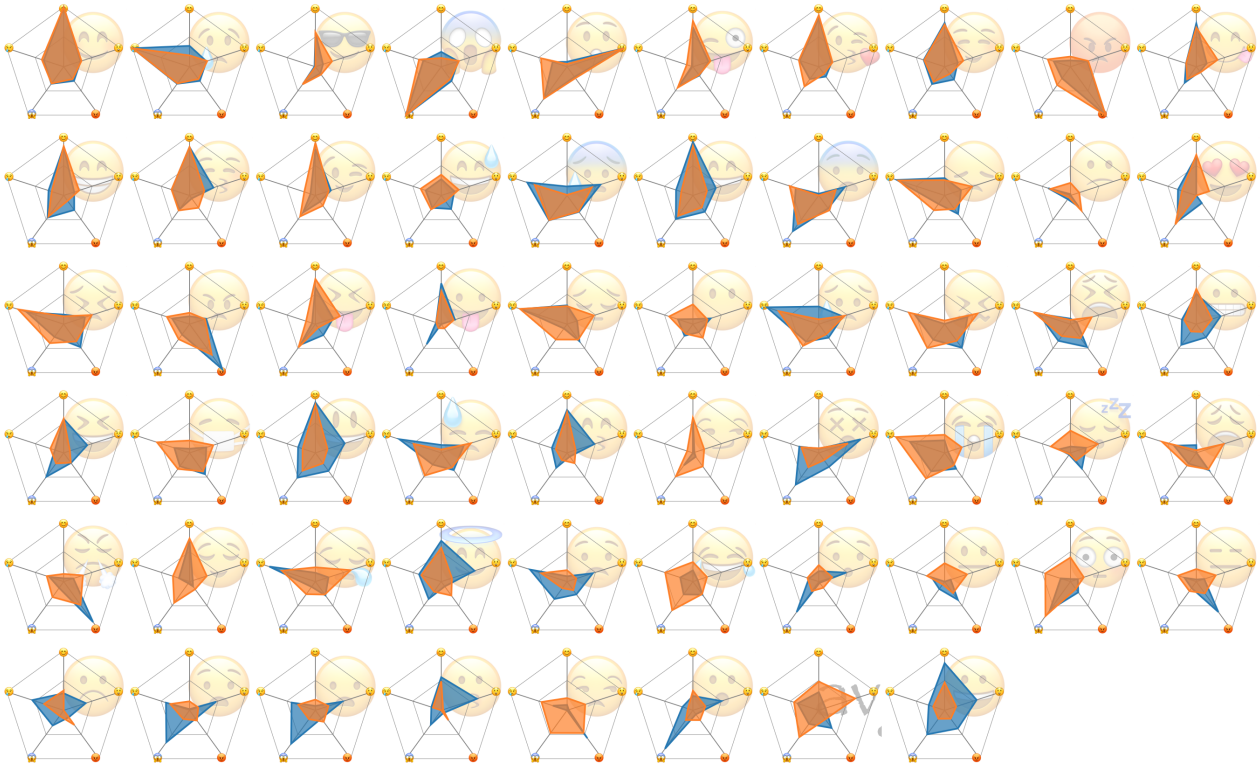

Figure 4: **The same projection,** but with data standardized over each axis.

These quantitative comparisons can be used to verify many psychological assumptions and claims. Before going into specific examples, these assumptions should be reiterated:

Claim 4. Our *Weibo* and *Twitter* datasets capture accurately cultural backgrounds of Weibo and Twitter users, respectively.

Claim 5. *Weibo* and *Twitter* users represent accurately native Chinese and English speakers, respectively.

3.4.1 Case Studies

Luo, et al., claimed that native Chinese people tend to decode facial expressions from the eyes.[7] Indeed, emojis with dotted (neutral, by design) eyes appear more neutral (read: closer to the middle pentagons) to *Weibo* users than to *Twitter* users.

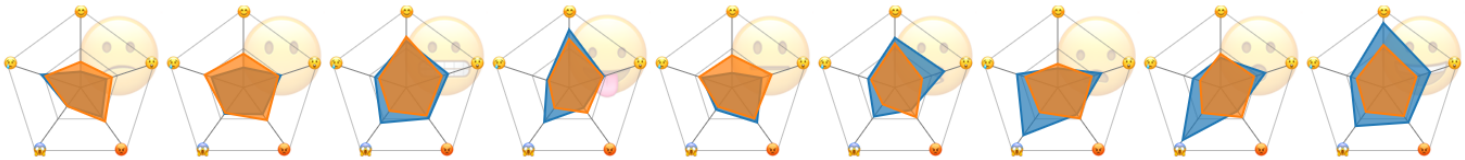

Figure 5: Emojis with dotted eyes appears more neutral to Chinese speakers.

Some other differences with respect to emoji interpretation between Chinese and English speakers include:

- When sending a 😒 emoji, an English speaker implies more a message of "I'm not scared at all" compared to Chinese people.
- On the contrary, with a 😯 emoji, Chinese users seem to be less frightened or surprised by the situation at hand.
- By a 😳 emoji, a *Twitter* user tends to be less happy about a particular issue, while a Chinese speaker implies no tendency with respect to happiness.

- This is the opposite when it comes to the 😟 emoji. While it conveys no information about the happiness level to a *Twitter* user, a *Weibo* user would type it to express unsatisfactoriness.

4 Conclusion

In this report, I have compared emotion inferences associated with 57 emojis across *Weibo* and *Twitter*. For this purpose, a pipeline is created to retrieve numerical projections of word vectors directly from textual corpora. Designed with generalization in mind, this pipeline enables inter-corpora comparison of projections from arbitrary tokens to some axial terms, provided that the axis are shared or unambiguously translatable across the training data. With this pipeline, cultural differences can be extracted from texts and revealed in a numerical manner. Novel to this field is the radar plot representation of such comparisons.

As a course project of CIS545, this paper focuses more on practicing techniques than conducting research in the most scientific manner. Many tools are involved in this process. Data cleaning is mostly done with a multi-node Spark environment in Python flavor, and vectorization is achieved by the *gensim* package. Model construction and further data manipulation is completed with *Pandas*. In the end, visualization is powered by *Matplotlib* and *D3.js*.

Acknowledgment

Datasets of archived microblogs, as well as computational resources, are kindly provided by the World Well Being Project (WWBP) at the University of Pennsylvania (UPenn). Hence, this paper is subject to disclosure policies at WWBP and at UPenn.

I'd like to express gratitude towards Prof. Lyle Ungar and Dr. Sharath Chandra Guntuku for guiding me through this project. Great thanks to Prof. Zachary Ives, whose lectures has always been pleasant to attend. Last but not least, my applause to Teaching Assistants and voluntary answerers on Piazza – their devotion to others is highly recognized and appreciated.

References

[1] Extracting, transforming and selecting features - Spark 2.3.0 Documentation.

[2] gensim: topic modelling for humans.

[3] SwiftKey Emoji Report | The United States | Data.

[4] How Emojis are Perceived Differently by Different Cultures, February 2018.

[5] Francesco Barbieri, Francesco Ronzano, and Horacio Saggion. What does this Emoji Mean? A Vector Space Skip-Gram Model for Twitter Emojis. page 6.

[6] Laurens van der Maaten and Geoffrey Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[7] Xiaoqin Mai, Yue Ge, Lin Tao, Honghong Tang, Chao Liu, and Yue-Jia Luo. Eyes Are Windows to the Chinese Soul: Evidence from the Detection of Real and Fake Smiles. *PLOS ONE*, 6(5):e19903, May 2011.

[8] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. page 9.

[9] Radim Řehůřek. Word2vec Tutorial.

[10] Kam-Fai Wong, Wenjie Li, Ruifeng Xu, and Zheng-sheng Zhang. *Introduction to Chinese Natural Language Processing*. Morgan & Claypool Publishers, November 2009. Google-Books-ID: DZReAQAAQBAJ.