The blog post should be “narrative” and interesting. It should not read like an academic paper. Read a some of the existing examples in the medium collection to get a sense of the voice and style. [https://medium.com/learning-journalism-tech (Links to an external site.)](https://medium.com/learning-journalism-tech)

The blog post should speak to the following:

* Brief description of the project and who it’s for.
* Why you chose this project.
* How you did it.
* What were the challenges/discoveries
* What’s next / Possible improvements.

**Team members: Dara Rubin, Diane Liu, Colin Mo, Chenhui Zhou**

**About emojify**

emojify is a web application that allows users to take English words or phrases and translate them into the emoji equivalent. emojify is meant to be a fun and different way for users to express their content. As digital media evolves and the landscape moves towards drawing a veritable amount of attention from millennials, utilizing a pure or mixed headline of emojis and traditional text may serve as a way for headliners to make users pause. By identifying that potential users have a desire for such a process and others may simply be interested in having fun with what we can create, we’ve provided an interface for the process in the most non-distracting way possible, emphasizing on the user’s ability to clearly get an output for an input, as well as share whatever funny results they end up with. Originally, our team focused on newspaper and blog post headlines, with the hopes that maybe a social media campaign manager would use our website to translate blog post or other titles before releasing them to attract millennials, or really anyone who likes emojis. emojify may also be for the trendy editor who wishes to attract a different kind of article viewership, or the bored office worker seeking a few minutes of entertainment. While millennials are certainly more adept at using emojis efficiently and correctly, their parents often find them amusing to send as well, whether they are successful or not. So in a sense, emojify can really be for anybody.

**Why we chose this project**

Many of us already used the occasional emoji in place of words and we thought it would be fun to take on translating an entire sentence or phrase into emojis and seeing how close we could get (and if people could understand it). “As a journalism major, it intrigues me when anything comes out that can have an impact on crafting news stories, as it is a formulaic construct that sees few amounts of innovation,” said Colin.

**How we did it**

Our app works by taking user input in plain English and using our Flask server. Here, the phrase goes through many parts - it finds the root of the word, trims punctuation, performs synonym lookups (via WordNet), handles negation, changes any capitalization to lowercase, and also looks at the words in groups of two (bigram lookup) since many of the emoji definitions (e.g. New York) have two words before doing a unigram lookup. All of this is based on a .json file we have with emojis, their definitions, and some synonyms. This file is lemmatized before it returns to the user. After all these tasks run, the emoji translation goes back through Flask and returns a string of emojis to the user on our website.

**Problems and Discoveries**

We encountered several problems along the way and learned a lot in the process. Our first problem was really just figuring out how to start: What emojis would we use? Would we create our own corpus or dictionary from scratch or could we find a good existing one? We decided on Apple’s emojis because they were the most accessible, although this caused some minor issues because our group members have both Macs and PCs which display the emojis differently. We also found an existing corpus with emojis, similar keywords and partial definitions; however, the dictionary needed a lot of manual work to fix keywords and expressions. Another early stumbling block our team encountered was figuring out how to go from a human interpretation of a phrase into more of a computational process that still makes sense to the user.

Since our product is so simple, when drawing the paper prototype early on we felt like we were missing something because we only had content for one page, but we realized that this was okay for what our group was trying to create.

Once we had the basics up-and-running and we began testing emojify, we ran into a new problem: what if a word we put in the translator doesn’t exist in our dictionary? There were only so many words we could add in manually. To solve this problem, any word that cannot be found gets run through WordNet, which looks up synonyms and usually finds something in our dictionary that matches, although it is not always the best match.

The backend would originally separate “New York” into “new” and “york”, providing different emojis compared to the ones we wanted, but that was fixed by first doing a bigram and then a unigram lookup for each phrase.

**What’s Next for emojify?**

Considering the lack of emojis for many words and phrases, we would love to have an expanded emoji library to utilize more translations in the future. The lack of many emojis for our translator is akin to translating into English without being able to use prepositional words or phrases; far too many semantics are lost in the translation process.

We would also need to consider the impact of Apple OS and non-Apple computers getting different icons corresponding to the same emoji; in particular, some emojis are only displayed in black and white on non-Apple products, while the Apple products display it in color. Depending on the context, sometimes this may have a factor for the receiver if a user on one device shared it and the context became different due to the difference in emojis on a different system.