CIS590/890 DL - Assignment 4

1 Goal

The goal of this assignment is to use and compare Word2vec and GloVe word embeddings in the context of a text classification problem, specifically, classification of short sentences with respect to five emojis, as described below.

2 Emoji Classifier

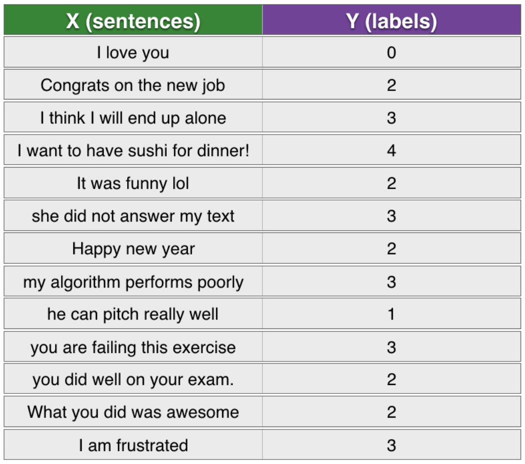
Your task is to use word vector embeddings to build an Emojifier.

Have you ever wanted to make your text messages more expressive? Your emojifier app will help you do that. So, rather than writing "Let's get together for lunch or coffee! Love you!" the emojifier can automatically turn this into "Let's get together for lunch 🍴 or coffee ☕! Love you❤️!"

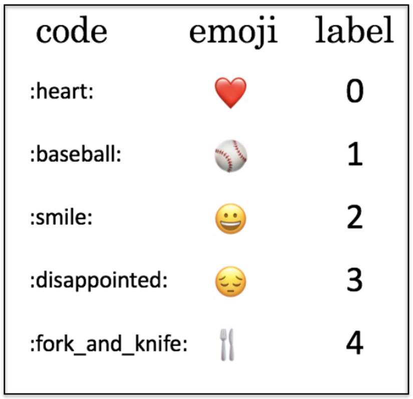
You will implement a classifier/network which inputs a sentence (such as "Let's go see the baseball game tonight!") and finds the most appropriate emoji to be used with this sentence (⚾). By using word vectors, you'll see that even if your training set explicitly relates only a few words to a particular emoji, your algorithm will be able to generalize and associate words in the test set to the same emoji even if those words don't even appear in the training set. This allows you to build an accurate classifier mapping from sentences to emojis, using just a small training set.

3 Dataset EMOJISET

You will use a small EMOJISET to build your classifier. The dataset consists of 127 sentences (strings). The label y of an instance/sentence x is an integer between 0 and 4 corresponding to an emoji for that sentence. The following figure shows some examples from the dataset, and their corresponding labels.



X (sentences)	Y (labels)
I love you	0
Congrats on the new job	2
I think I will end up alone	3
I want to have sushi for dinner!	4
It was funny lol	2
she did not answer my text	3
Happy new year	2
my algorithm performs poorly	3
he can pitch really well	1
you are failing this exercise	3
you did well on your exam.	2
What you did was awesome	2
I am frustrated	3

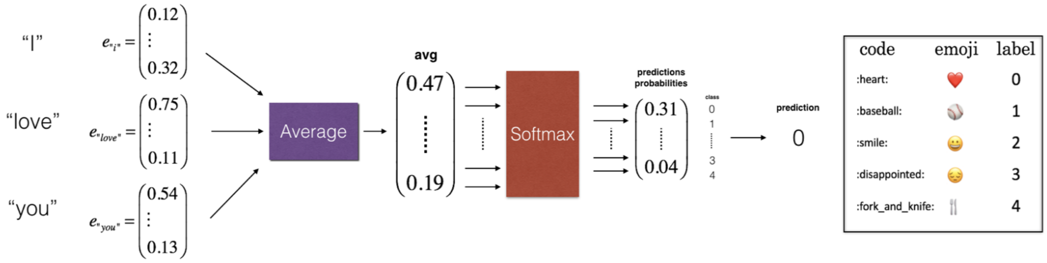


code	emoji	label
:heart:	❤️	0
:baseball:	⚾	1
:smile:	😀	2
:disappointed:	😔	3
:fork_and_knife:	🍴	4

The dataset has been split into training and test subsets, to be used for training and evaluating the classifier, respectively.

4 Overview of the Emojifier

The input of the model is a string corresponding to a sentence (e.g., "I love you"). This string is represented as the average of the embeddings of the words in the sentence. In the code, the output will be a probability vector of shape (1,5), that you then pass in an argmax layer to extract the index of the most likely emoji output. The architecture of the model is shown below:



"I"
$e_{\text{"I"}} = \begin{pmatrix} 0.12 \\ \vdots \\ 0.32 \end{pmatrix}$
"love"
$e_{\text{"love"}} = \begin{pmatrix} 0.75 \\ \vdots \\ 0.11 \end{pmatrix}$
"you"
$e_{\text{"you"}} = \begin{pmatrix} 0.54 \\ \vdots \\ 0.13 \end{pmatrix}$
Average
avg
$\begin{pmatrix} 0.47 \\ \vdots \\ 0.19 \end{pmatrix}$
Softmax
predictions probabilities
$\begin{pmatrix} 0.31 \\ \vdots \\ 0.04 \end{pmatrix}$
class
0
1
3
4
prediction
0
code emoji label
:heart: ❤️ 0
:baseball: ⚾ 1
:smile: 😀 2
:disappointed: 😔 3
:fork_and_knife: 🍴 4

In addition to average word embeddings, we will also experiment with min and max word embeddings (by taking element-wise min or max over the word embedding vectors).

5 Word Embeddings

We will experiment with pre-trained Word2vec and GloVe embeddings.

Word2vec embeddings trained on Google News are available at `https://code.google.com/archive/p/word2vec/`, specifically in the archive GoogleNews-vectors-negative300.bin.gz. The embeddings have dimension 300.

GloVe embeddings are available at `https://nlp.stanford.edu/projects/glove/`. You will experiment with GloVe word embeddings in glove.6B.zip (trained from Wikipedia 2014 + Gigaword 5), and also word embeddings in glove.twitter.27B.zip (trained on a Twitter corpus). Each set contains a variety of dimensions.

6 Tasks

Perform the following tasks:

1. Convert sentences to average word embeddings (after converting every sentence to lower-case, and splitting the sentence into a list of words). Alternatively, convert sentences to min and max word embeddings (by taking element-wise min or max over the word embedding vectors).

2. Use GloVe embeddings of size 50 (trained on Twitter) to train and evaluate the classification model described above. Show the confusion matrix. What is the accuracy of the classifier on the test dataset? Show 10 examples that are correctly classified by the model trained, and 10 examples that are misclassified by the model. Are the results surprising at all? Can you explain the misclassifications?

3. Using the same GloVe embeddings, train models that use min and max word embeddings (as opposed to average embeddings). Compare the results (accuracy) of the models obtained with avg/min/max sentence representations, respectively. Which representation gives the best results? Do the results improve if you use avg/min/max together (in this case, the sentence vectors will be 3x50)?

4. Use 300d Word2vec and 300d Glove embeddings (trained on Wikipedia) to train the model with the average word embedding representation for sentences. Between 300d Word2vec embeddings and 300d GloVe embeddings, which set performs better?

5. Train models using the 50d, 100d, and 200d Glove embeddings using both the set trained on Wikipedia and the set trained on Twitter. For each set, what embedding dimension gives better results? Between the two sets (Wikipedia versus Twitter), which one performs better?

7 What to submit

Submit a Jupiter Notebook containing your code and results/discussion, or python code together with a report file showing the results/discussion for different task.