Natural Language Processing
Phase №3: Final

Pooya Kabiri

Department of Computer Science

Iran University of Science and Technology

July 2021

Contents

List of Figures **1**

1 Word2Vec **2**

2 Tokenization **3**

3 Dependency Parsing **3**

4 Language Model **4**

5 Fine-Tuning **5**

List of Figures

1 Cosine Similarity 2
2 unk percent, for vocab size 3

1 Word2Vec

I used A2 code on two class of my data which are chandler and phoebe.

I changed the code for StanfordSentiment Dataset and turned it into Friends-Dataset which I could use with my own data. I trained a word2vec model for each class, each for 5000 iterations, and saved the final models as requested.

For every common word between the two classes I computed a cosine similarity, which was 2640 common words, every 20 of them in a single report photo.

	Word	Cosine Similarity
0	community	0.39232285411260005
1	heard	0.3780190594343536
2	brought	0.37398772581463163
3	layer	0.45830644143432026
4	crush	0.40704597840492185
5	nervous	0.4087547362407764
6	your	0.42251687845246166
7	wine	0.3432563343645686
8	slip	0.3458676964242077
9	calling	0.4271860879747031
10	were	0.382871473394361
11	transferred	0.31540630467124703
12	resolution	0.3837279687318935
13	blonde	0.3544130959042138
14	sebastian	-0.16356062866300088
15	die	0.32176441927151905
16	yum	-0.16516395158017447
17	helping	0.40658989532894513
18	useless	0.44120407594566374
19	shall	-0.25995695629021964

Figure 1: Cosine Similarity

For example in this photo, the word 'yum' has a low similarity, and I think that because Phoebe is a very sarcastic person and usually uses the word 'yum' in a sarcastic way, meaning the opposite (not delicious), but chandler is not sarcastic and loves food, so that's maybe why the similarity in the context of the dialogues very low.

In this photo, the word 'Janice' has a high sim score, they both hate her character and usually everything stated about Janice is negative by the both of them. They both hate her so much. I can think of this as a reason that maybe Janice has a high sim score in both characters' context.

There are 132 report pictures under `reports/word2vec/` and it contains sim score for all common vocabulary.

2 Tokenization

for this section I used SentencePiece module as requested, and trained data for each 2 labels separately.

On each training procedure, 0.2 of the data is used for testing and counting the number of predicted unk *tokens*.

the maximum number for vocabulary which the framework allowed me to use was 11737.

Figure 2: unk percent, for vocab size

As seen from the table, the more the vocab size, the tokenizer performs better and the percentage of unk tokens is smaller. Best result is from the maximum possible size of vocab (11737) , which is 1.59%, and also the size is not too big and it can be used without any size problems. This model is copied under `models/tokenization` as the best model.[1]

3 Dependency Parsing

For this part I used A3 code, trained it on the assignment's train data. I manually parsed 10 sentences from my data using Conllu editor, I did POS tagging and Dependency Parsing on them, and then fed them into the neural network as test data. I got a UAS score of 76.42% on these 10 sentences.

The annotated file for 10 test sentences can be found under `data/parsing`.

Example №1:
My erection is back !

UAS score for this example : 0.5

The model has chosen 'erection' as the root, but it is false. The head for 'erection' is 'is', but the model has predicted it the other way.

Example №2:
I learned never to borrow money from friends .

UAS score for this example : 0.875, which is pretty good.

The model has chosen 'learned' for the head of 'never', instead of the correct dependency which is that 'borrow' is the correct head for 'never'. In this

[1] Used resource

case I think because 'never' is an adverb and there are two verb phrases in the sentence, it is hard to predict that this adverb describes which of the verbs.

Example №3:
I tend to keep talking until somebody stops me .

UAS score for this example : 0.88, which is also decent.
The model only predicted one dependency wrong, which is the head of 'stops'. Model prediction is 'tend', but the correct head is 'keep'. The model didn't predict that this adverb clause belongs to which verb.

Example №4:
My idea of a vacation does not involve spmething sucking on my nipples until they are raw .

UAS score for this example : 1.0
The model predicted this long sentence fully correctly. It is a funny, informal sentence but the model predicted all of the dependencies correctly, resulting in a score of UAS 1.0. (Sorry for the adult language :D).

4 Language Model

For this section I used a code from the provided source. The model is a LSTM with 3 layers, hidden state size of 128, and also embedding size of 128, followed by a dropout layer (p=0.2).
the data fed to this network is cleaned using stopword and contradiction removal and tokenization.
I used the two models to generate sentences providing the model a single starting word. the full results are under `reports/language_model`.

coffee spade the plate in up out guitar . so i	Phoebe
coffee sooner laughs tomorrow , but uh beam look you said	Chandler

smelly focusing . how just i m , and very coffee	Phoebe
smelly battling piece open feels time we do ? i know	Chandler

monica dolls jerks spray him . hi ! it s me	Phoebe
monica just chicken kind than individual vaporizing teach you the in	Chandler

joey smoke got lessons one . because , and cool .	Phoebe
joey rehearse ! thanks there interview ? yes , i think	Chandler

home latour base room ? because because you just get us	Phoebe
home howie staying , take this then . and	Chandler

The result are not satisfying, but I can see some very little changes in the generated texts. Phoebe usually speaks in a very word-by-word way and uses very short sentences.[2]

5 Fine-Tuning

For this section, I used a code from HuggingFace which is provided in the footnote. It used a model checkpoint of distil GBT2 for 'casual language modeling' task. I used this pretrained model to fine tune two labels of my data, which are chandler and phoebe, as before.

I splitted the data in a 80-20 way for train and dev sets. I trained the model for 5 epochs.

The perplexity for each class:

Chandler	43.57
Phoebe	40.59

Now, I used the same start words as the section 4 for text generation. Here are the results:

coffee !um , so , are you listening ?oh , how is it ?really ?no	Phoebe
coffee and honey , he gave us .we have you all together in this apartment !they	Chandler

smelly and the man !hey !why mma mike mike , i know !hey	Phoebe
smelly !good bye mary !oh hey , i'm .well , this is what i	Chandler

monica , it would be great !oh !yeah :no ?i am still here ;so	Phoebe
monica : that i said .look what , i am here .it sucks to have	Chandler

joey , well yeah .i love you !she can take in food she knows .i	Phoebe
joey !i never got into sex with her .i just wanted to get in the way	Chandler

home was there ?she has to make that happen .but then we have to say we don t	Phoebe
home aww , they can get everything .did that .oh hey , now i should do this	Chandler

The results are much better improved in contrast to section 4. The sentence are correct and meaningful, just the combination of them doesn't make sense, but each sentence has a better overall structure. [3]

Due to Github Limitations, I uploaded my final models on Google Drive, and you can access it here.

[2] Used Code
[3] Used Code