

**In the name of God**



Department of Computer Engineering

# **Natural Language Processing**

## **Final Phase Report \***

**Yeganeh Morshedzadeh**

Student Number: 96521488

Spring 2021

---

\*<https://github.com/yegmor/NLPPProject>

# Contents

I	Word2Vec	6
II	Tokenization	8
III	Parsing	9
IV	Language Model	12
V	Fine-tuning	14

## List of Figures

1	t-SNE visualization for man . . . . .	7
2	Universal Dependencies for 10 sentences . . . . .	10
3	Examples for simple language model . . . . .	13
4	Training bert classification language model . . . . .	14
5	Evaluating bert classification language model . . . . .	14
6	Examples for bert classification model . . . . .	15
7	Examples for distilgpt2 language model . . . . .	16

## List of Tables

1	Word2Vec vocabulary size . . . . .	6
---	------------------------------------	---

## **Abstract**

In this project, we tried to use Natural Language Processing to better understand Depression and Anxiety posts. The dataset is gathered from Reddit communities [r/depression](#) and [r/Anxiety](#).

For this project, at first, we wrote a project proposal ([Google Docs](#)), and afterwards, in the first phase ([Google Docs](#)), we gathered data and made some exploratory data analysis.

In the final phase, we went deeper, and tried various NLP tasks, such as, computing Word2Vec, Tokenization, Parsing, and creating a language model based on our the dataset.

## Part I

# Word2Vec

Filename: 3\_word2vec.ipynb

## Code

We use Gensim implementation of word2vec: <https://radimrehurek.com/gensim/>

For this part we have three Word2Vec models, named as `dep_w2v_model`, `anx_w2v_model`, and `all_w2v_model`. Moreover, with boolean parameters, `load` and `save`, the model will be saved and/or loaded in the `my_word2vec` function.

Table 1: Word2Vec vocabulary size

	label	vocab_size
0	depression	2054
1	anxiety	2175
2	all	3223

## Results and Examples

In this part, we used t-SNE visualization. t-SNE is a non-linear dimensionality reduction algorithm that attempts to represent high-dimensional data and the underlying relationships between vectors in a lower-dimensional space.

To make the visualizations more relevant, we will look at the relationships between a query word (in **\*\*red\*\***), its most

similar words in the model (in **blue**), and other words from the vocabulary (in **green**)

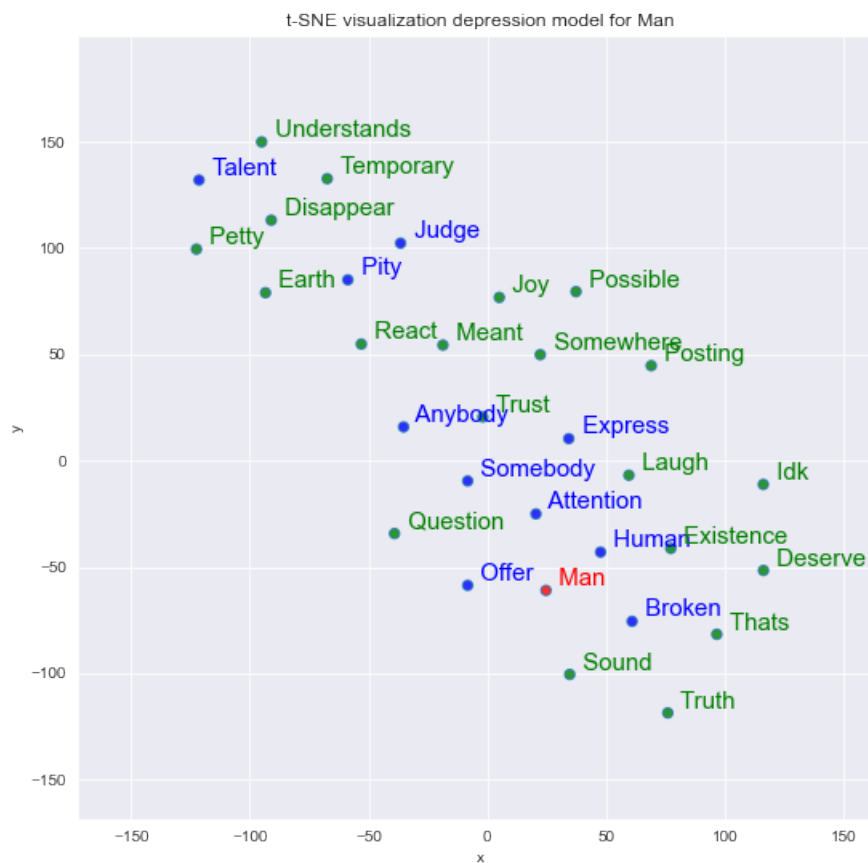


Figure 1: t-SNE visualization for man

## Part II

# Tokenization

**Filename:** 4\_tokenization.ipynb

## Code

In this part, we have used KFold to split our data into train and test. Afterwards, we train SentencePiece model based on the data. Lastly, we compute <UNK> on our test dataset.

## Results and Examples



## Part III

# Parsing

In this part, we used Stanza, which is a Python NLP Package, and a collection of accurate and efficient tools for the linguistic analysis of many human languages. Starting from raw text to syntactic analysis and entity recognition, Stanza brings state-of-the-art NLP models to languages of your choosing.

More specifically, we used their [Online Demo](#) to create a manual .CoNLL file based on our dataset. Later, we can use [Universal Dependencies CoNLL viewer](#) to automatically generate parse tree from .CoNLL file.

## Results and Examples

The reported Unlabeled Attachment Score (UAS) on our test file was 86.05.

## Universal Dependencies:

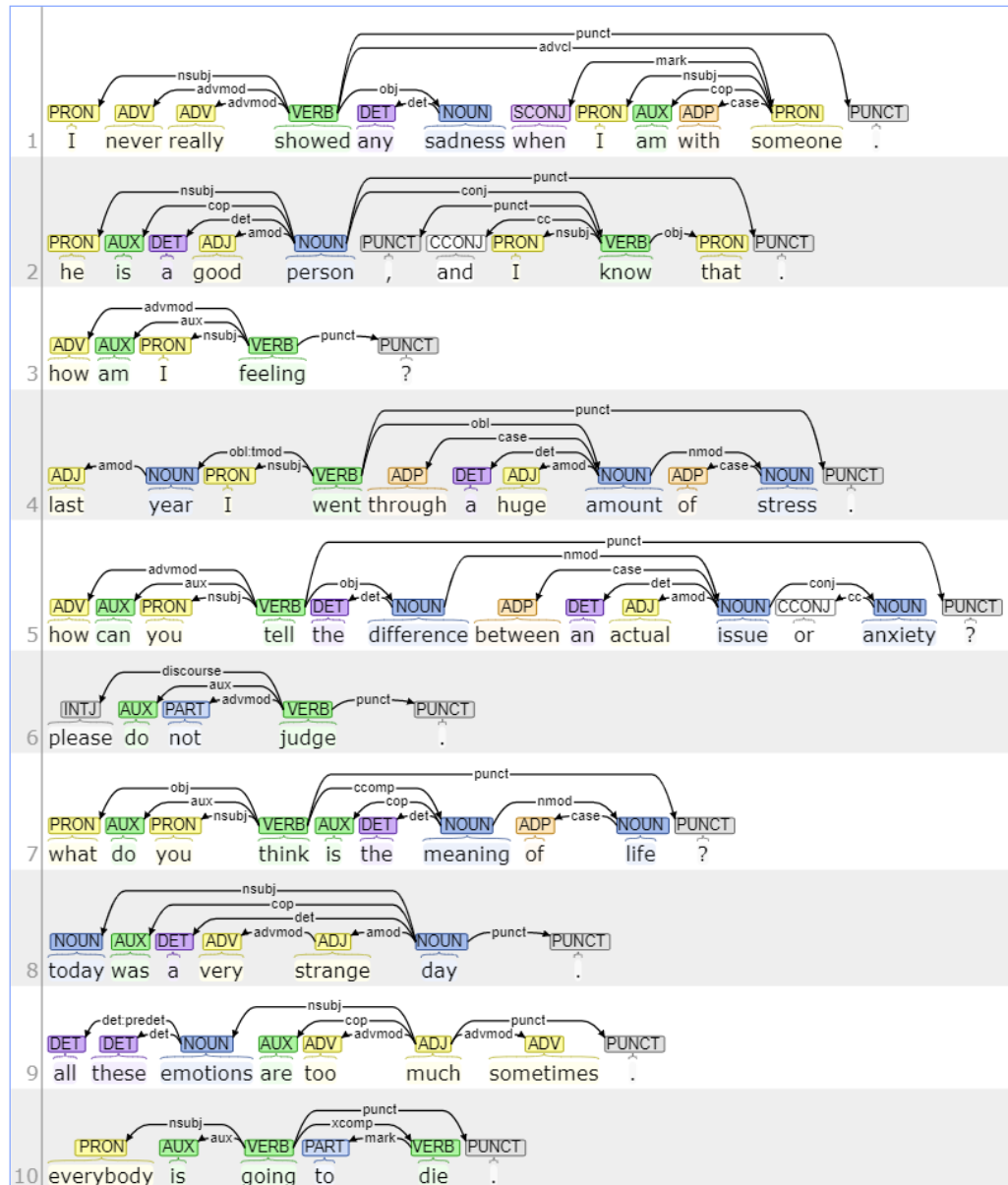


Figure 2: Universal Dependencies for 10 sentences

1. I never really showed any sadness when I am with someone.
2. he is a good person, and I know that.
3. how am I feeling?

4. last year I went through a huge amount of stress.
5. how can you tell the difference between an actual issue or anxiety?
6. please do not judge.
7. what do you think is the meaning of life?
8. today was a very strange day.
9. all these emotions are too much sometimes.
10. everybody is going to die.

## Part IV

# Language Model

**Filename:** 5\_language-model.ipynb

In this part, we have develop a model of the text that we can then use to generate new sequences of text.

## Code

We will start by preparing the data for modeling. We have picked a length of 50 words for the length of the input sequences, and we have attached 1 output word at the end of the sequences. Afterwards, we save the long list of sequences.

For our language model, after loading our dataset, we used an Embedding Layer to learn the representation of words, and a Long Short-Term Memory (LSTM) recurrent neural network to learn to predict words based on their context. More precisely, we have used a two LSTM hidden layers with 100 memory cells each. Following, we have a dense fully connected layer with 100 neurons connects to the LSTM hidden layers to interpret the features extracted from the sequence.

After the model is trained, we use `generate_seq` function that takes as input the model, the tokenizer, input sequence length, the seed text, and the number of words to generate.

## Results and Examples

	seed_text	generated	is_anxiety	n_seq_words
0	firmly believe social anxiety sole reason deal...	enjoy water mate major gave center shower deal...	0	20
1	much ha happened last year covid ha definitely...	away feel like shallowest blandest person thin...	0	20
2	think everyone understands fear missing fomo u...	neighbor passionate therapy deeper word trigge...	1	20
3	took first dose lexapro today doctor advised s...	anxiety fear thought way something wa talking ...	1	20

Figure 3: Examples for simple language model


## Part V

# Fine-tuning

## Classification

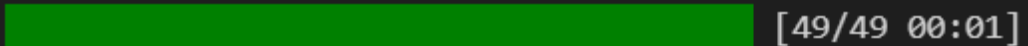
Filename: 6\_finetime\_classification.ipynb

## Code



Epoch	Training Loss	Validation Loss	Runtime	Samples Per Second
1	No log	0.446038	1.187500	325.040000
2	0.606900	0.370412	1.188600	324.739000
3	0.374500	0.347635	1.196000	322.729000
4	0.294500	0.534144	1.199100	321.916000
5	0.227300	0.571521	1.201800	321.177000
6	0.121500	0.631738	1.205400	320.236000
7	0.088700	0.716783	1.207600	319.642000
8	0.062300	0.753405	1.210900	318.777000
9	0.048100	0.778459	1.210700	318.820000
10	0.041200	0.779626	1.211300	318.662000

Figure 4: Training bert classification language model



```
{'eval_loss': 0.779626190662384,  
'eval_accuracy': 0.8730569948186528,  
'eval_runtime': 1.2255,  
'eval_samples_per_second': 314.975}
```

Figure 5: Evaluating bert classification language model

## Results and Examples

	seed_text	predicted	actual
0	tired tired fighting live care anything tired ...	0	0
1	know one perfect life bad like literally every...	0	0
2	17 ive always attracted woman think pocd destr...	1	1
3	love sport play never coached never played spo...	1	1

Figure 6: Examples for bert classification model

## Language Model

**Filename:** 7\_finetune\_language-model.ipynb

In this part, we have used our Reddit dataset to fine-tune our distilgpt-2.

For this purpose, we first prepare the dataset and build a TextDataset, load the pre-trained GPT-2 model and tokenizer, initialize Trainer with TrainingArguments, and finally, train and save the model.

### Code

The TextDataset is a custom implementation of the Pytorch Dataset class implemented by the transformers(4.2.2) library. Also, we have used the tokenzier from the distilgpt-2 model on huggingface.

Afterwards, we initialize Trainer class that provides an API for feature-complete training, and we set the Hyperparameters we are going to use in the TrainingArguments.

## Results and Examples

	seed_text	is_anxiety	generated_depression	generated_anxiety
0	talking family member face face friend gave bad ne	0	talking family member face face friend gave bad nek think bad one see world face bad get worse every time friend find fault find family friend give wrong friend feel bad day get worse want someone think really like world like see world like see past see world like	talking family member face face friend gave bad nexie irlfax prescription day ago ago bought caffeine drink caffeine med last night thought wa really bad ever tried caffeine drink really help feel good still got depressed since first thought wa really bad wa much way
1	stuck life want succeed taking much losing sad ene	0	stuck life want succeed taking much losing sad enecember last 2nd year lost love hope friend wa im going get better i dont remember last month wa looking im thinking maybe one day would start asking thing one day try get therapy everyday one night want	stuck life want succeed taking much losing sad eneapnity got good time anxious make anxiety worse day would get bad experience bad experience even worse go see doctor day think something anxiety make mistake stop going see doctor try trying try every day try find
2	turned kind long post tl dr recently depression an	1	turned kind long post tl dr recently depression an average day last 5 minute yesterday started dating would rather never get married got really lonely got divorced 2 year ago 4 year ago 2 year ago decided to return get married 2 year ago 1 year	turned kind long post tl dr recently depression an 11 year old went doctor wanted 2 month ago 3 month ago 4 month ago also went 3 month ago back last year ha diagnosed anxiety disorder later diagnosis anxiety disorder anxiety disorder schizophrenia diagnosis
3	issue forever whenever something bad happens make	1	issue forever whenever something bad happens make much worse time wa time every time time like want want feel like need something else could live without always felt like really bad time want become useess one time think like wrong something	issue forever whenever something bad happens make feel alone feel alone never want to deal honestly feel alone feel like getting together like never felt alone feel like really hard feel like need help someone know anyone else always scared really

Figure 7: Examples for distilgpt2 language model



## References

- [1] <https://towardsdatascience.com/goodbye-world-4cc844197d51>
- [2] [https://colab.research.google.com/github/google/sentencepiece/blob/master/python/sentencepiece\\_python\\_module\\_example.ipynb#scrollTo=ee9W6wGnVteW](https://colab.research.google.com/github/google/sentencepiece/blob/master/python/sentencepiece_python_module_example.ipynb#scrollTo=ee9W6wGnVteW)
- [3] [https://gmihaila.github.io/tutorial\\_notebooks/gpt2\\_finetune\\_classification/](https://gmihaila.github.io/tutorial_notebooks/gpt2_finetune_classification/)
- [4] [https://colab.research.google.com/github/philschmid/fine-tune-GPT-2/blob/master/Fine\\_tune\\_a\\_non\\_English\\_GPT\\_2\\_Model\\_with\\_Huggingface.ipynb](https://colab.research.google.com/github/philschmid/fine-tune-GPT-2/blob/master/Fine_tune_a_non_English_GPT_2_Model_with_Huggingface.ipynb)
- [5] [https://github.com/huggingface/notebooks/blob/master/examples/language\\_modeling.ipynb](https://github.com/huggingface/notebooks/blob/master/examples/language_modeling.ipynb)
- [6] <https://www.kaggle.com/pierremegret/gensim-word2vec-tutorial>
- [7] <https://machinelearningmastery.com/how-to-develop-a-word-level-neural-language-model-in-keras/>
- [8] <https://huggingface.co/transformers/training.html>
- [9] <https://www.kaggle.com/achintyatripathi/gensim-word2vec-usage-with-t-sne-plot>
- [10] <https://colab.research.google.com/github/borisdyma/huggingtweets/blob/master/huggingtweets-demo.ipynb>
- [11] [https://huggingface.co/transformers/custom\\_datasets.html](https://huggingface.co/transformers/custom_datasets.html)

[12] [https://colab.research.google.com/github/google/sentencepiece/blob/master/python/sentencepiece\\_python\\_module\\_example.ipynb#scrollTo=-k5KbVgiYae-](https://colab.research.google.com/github/google/sentencepiece/blob/master/python/sentencepiece_python_module_example.ipynb#scrollTo=-k5KbVgiYae-)