M908 NLP

Mini Project: Stress Detection

Report

Christina-Theano (Theatina) Kylafi

LT1200012

Table of Contents

Introduction

Task Description

In the assignment at hand, the task of **stress detection** is implemented, researched and experimented on. The goal was to conduct a binary text classifier comparative research along with feature and model architecture experiments. A reddit social media stress dataset was selected and based on the lexical, syntactical and social media features provided plus additional ones from external datasets/models, it was used for examining the sensitivity / performance of multiple classification algorithms and artificial Neural Networks, in conjunction with different combinations of feature types.

Dataset

The initial dataset used is "**Stress Analysis in Social Media**", from the paper "**Dreaddit: A Reddit Dataset for Stress Analysis in Social Media**". In addition to the feature comprising it, emotion and depression prevailing class and score were added and tested on their contribution to stress detection, using distilbert and roberta pretrained models.

Emotion labels:

1. Sadness
2. Joy
3. Fear
4. Love
5. Anger
6. Surprise

Depression labels:

1. Not depression
2. Moderate
3. Severe

Deep Learning Frameworks

Depending on the available reference and the level of abstraction needed for each experiment, Keras, PyTorch and Tensorflow were used for the task implementation.

Structure

The .zip file attached, includes also:

1. The python notebook with the code that performed the experiments
2. Log files including the description of the models, their detailed scores in multiple performance measures, as well as comparative figures in order to come into conclusions about patterns, parameters, etc.
3. The final datasets created, with the additional features (emotion, depression) and the embeddings (BERT, Word2Vec)
4. Some dataset visualization figures

Background

The initial dataset, as the paper "**Dreaddit: A Reddit Dataset for Stress Analysis in Social Media"** suggests, consists of multiple **lexical**, **syntactic** and **social media** features:
" . . . we include features in three categories:

1. **Lexical** features. Average, maximum, and minimum scores for pleasantness, activation, and imagery from the Dictionary of Affect in Language (DAL) (Whissel, 2009); the full suite of 93 LIWC features; and sentiment calculated using the Pattern sentiment library (Smedt and Daelemans, 2012).

2. **Syntactic** features. Part-of-speech unigrams and bigrams, the Flesch-Kincaid Grade Level, and the Automated Readability Index.

3. **Social media** features. The UTC timestamp of the post; the ratio of upvotes to downvotes on the post, where an upvote roughly corresponds to a reaction of "like" and a downvote to "dislike" (upvote ratio); the net score of the post (karma) and the total number of comments in the entire thread under the post . . ."

Linguistic Inquiry and Word Count (LIWC)

As explained in its **manual**, the LIWC2015 Dictionary is the **heart of the text analysis strategy**. The default LIWC2015 Dictionary is composed of almost 6,400 words, word **stems**, and select emoticons. Each dictionary entry additionally defines one or more word categories or subdictionaries. LIWC2015 accesses a single text file, a group of files, or texts within a spreadsheet and analyzes each sequentially. For each file, LIWC2015 reads one target word at a time. As each target word is processed, the dictionary file is searched, looking for a dictionary match with the current target word. If the target word is matched with a dictionary word, the appropriate word category scale (or scales) for that word is incremented. As the target text file is being processed, counts for various structural composition elements (e.g., word count and sentence punctuation) are also incremented.

For each text file, approximately **90 output variables** are written as one line of data to an output file. This data record includes the file name and **word count**, 4 **summary** language variables (analytical thinking, clout, authenticity, and emotional tone), 3 general **descriptor** categories (words per sentence, percent of target words captured by the dictionary, and percent of words in the text that are longer than six letters), 21 standard **linguistic** dimensions (e.g., percentage of words in the text that are pronouns, articles, auxiliary verbs, etc.), 41 word categories tapping **psychological** constructs (e.g., affect, cognition, biological processes, drives), 6 **personal concern** categories (e.g., work, home, leisure activities), 5 informal language **markers** (assents, fillers, swear words, netspeak), and 12 **punctuation** categories (periods, commas, etc).

Dictionary of Affect in Language (DAL)

The Whissell Dictionary lets you look at the **mood of speech** in situ, as it is in the **person's mind**. Of course, it's not perfect, but for the spoken word (once it's transcribed), it allows a **measure of the person's state of mind** without using questionnaires, brain imaging, or biological tests.

The scores for **pleasantness** range from unpleasant (1) to pleasant (3).

The average for pleasantness in spoken English is 1.85 (with a standard deviation of .36).

The scores for **activation** range from passive (1) to active(3).

The average for activation in spoken English is 1.67 (with a standard deviation of .36).

The scores for **imagery** range from 1 (word evokes no image) to 3 (word evokes an image very easily).

The average for Imagery in spoken English is 1.52 (with a standard deviation of .63).

Because fewer 'raters' were used to establish an average value for how easily the word called an image (Imagery) to mind, the values for Imagery are only given to two decimal places.

Pattern Library for Natural Language Processing

Pattern is an extremely useful library in Python, that can be used to implement Natural Language processing tasks. The pattern is an open source, and free for anyone to use. It can be used for Text Mining, NLP, and Machine Learning. Sentiment Analysis is the application of Text Analytics that deals with understanding emotions and human sentiment from text data. It can be used to **understand opinions** or views expressed by a particular text.

The function in Pattern returns **polarity** and the **subjectivity** of a given text.
The **Polarity** result ranges from highly Positive to highly negative (1 to -1)
The **Subjectivity** ranges from 0 (Objective) to 1 (Subjective).

Flesch-Kincaid Grade Level

The Flesch Kincaid Grade Level is a widely used readability formula which assesses the approximate **reading grade level** of a text.

Automated Readability Index

The automated readability index (ARI) is a readability test for English texts, designed to gauge the **understandability** of a text.

Data Pre Processing

Text Embeddings

Two main sentence embeddings were tested:

1. **BERT** hidden state output embeddings of dimension **768** (based-uncased).

2. **Word2Vec** Google News pre-trained embeddings of dimension **300** (Only ~5% of the training set consists of unknown words and ~10% of the test set set – "UNK" was set to a zero numpy array in such cases, after ensuring that such a vector is not used for another token embedding).

Feature Selection

Except for the experiments on the text embeddings type and size, feature selection was also an interesting subject to research, therefore the performance and score of the following feature combinations were examined:

1. **All** numerical features (lexical, syntactic) from the initial dataset plus the emotion / depression score and categorical features (only)
2. **All** numerical features + text embeddings
3. Only **LIWC** lexical features
4. **LIWC** lexical features + text **embeddings**
5. Only **DAL** lexical features
6. **DAL** lexical features + text **embeddings**
7. Only text **embeddings**
8. Only **lexical** features (LIWC, DAL, sentiment)
9. **Lexical** features + text **embeddings**
10. Only **emotion** features (emotion class+score, depression class+score, sentiment, lexical feats related to emotion)
11. **Emotion** features + text **embeddings**

Data Visualization

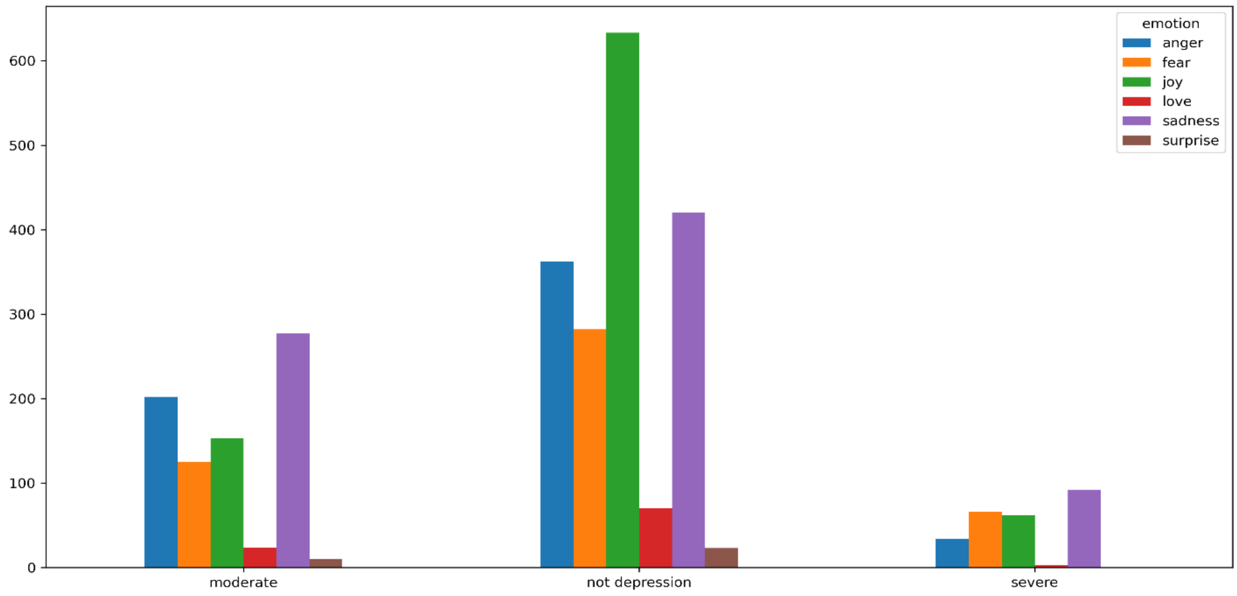


Figure 1. Depression – Emotion status

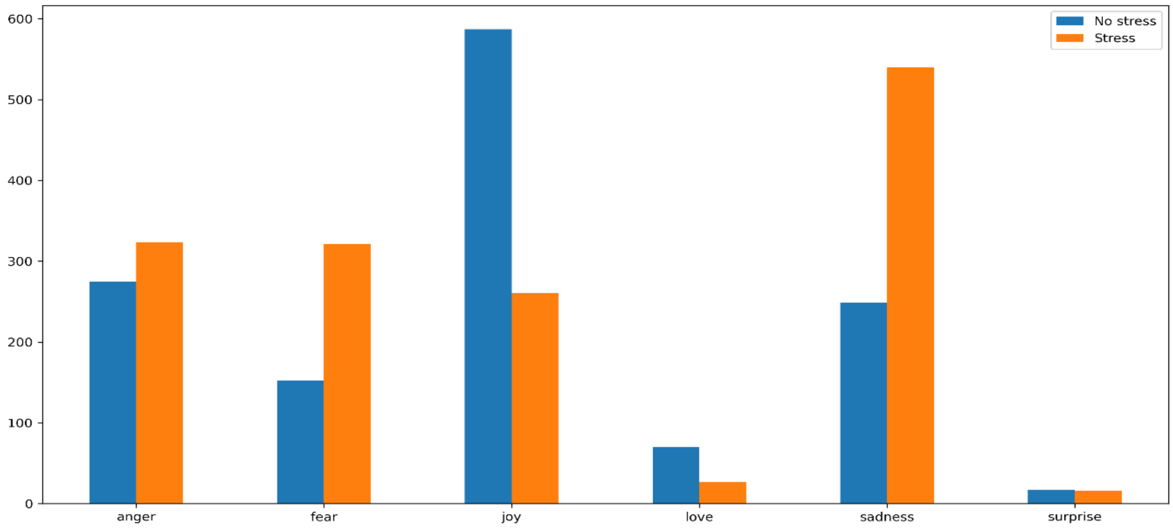


Figure 2. Emotion – Stress label

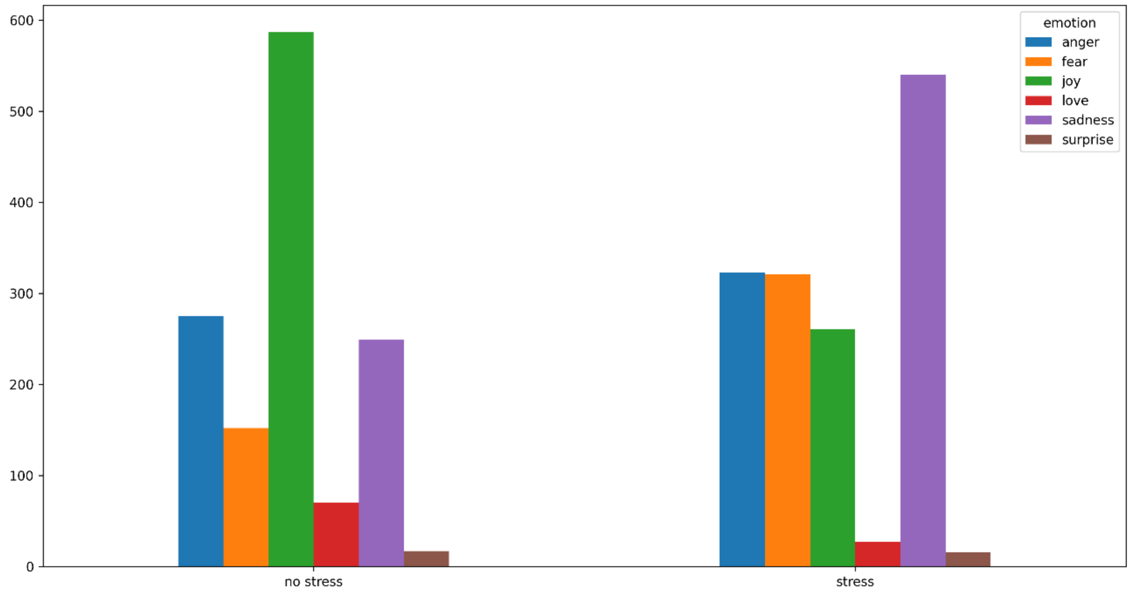


Figure 3. Stress label – Emotion graph

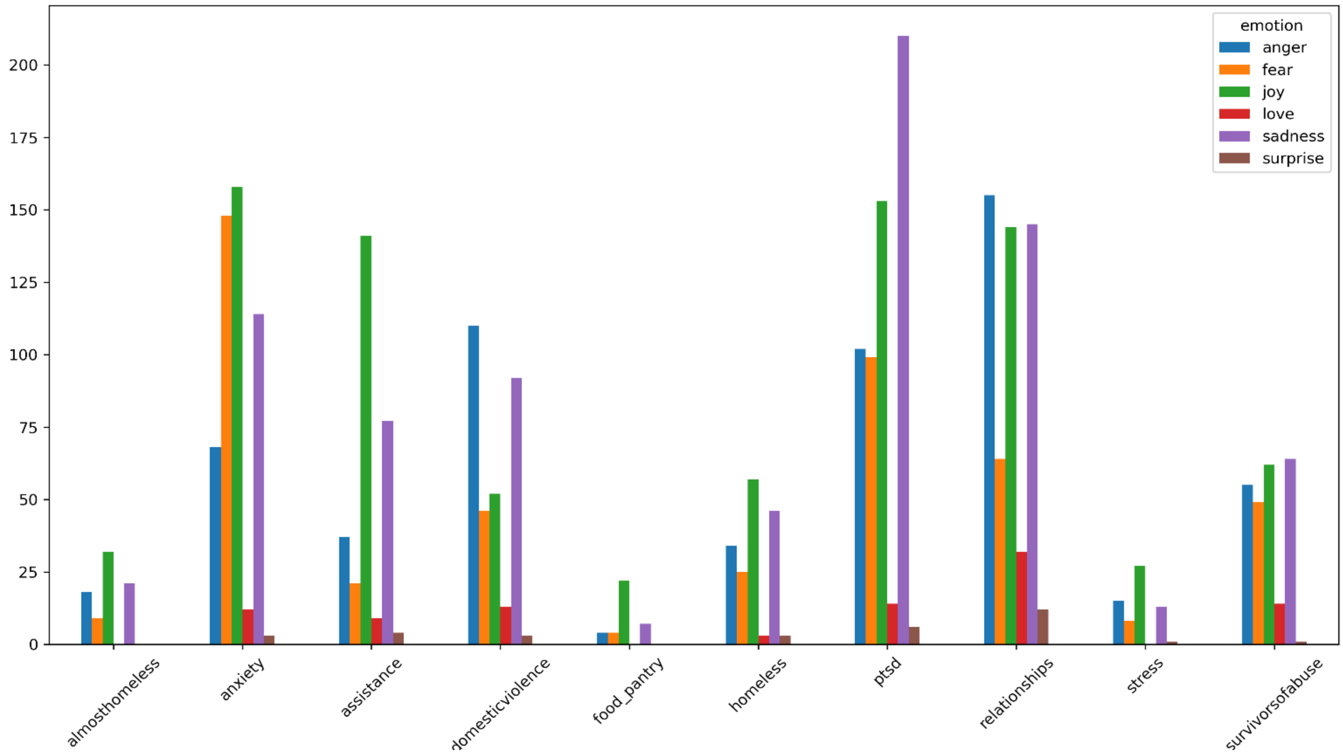


Figure 4. Subreddit – Emotion graph

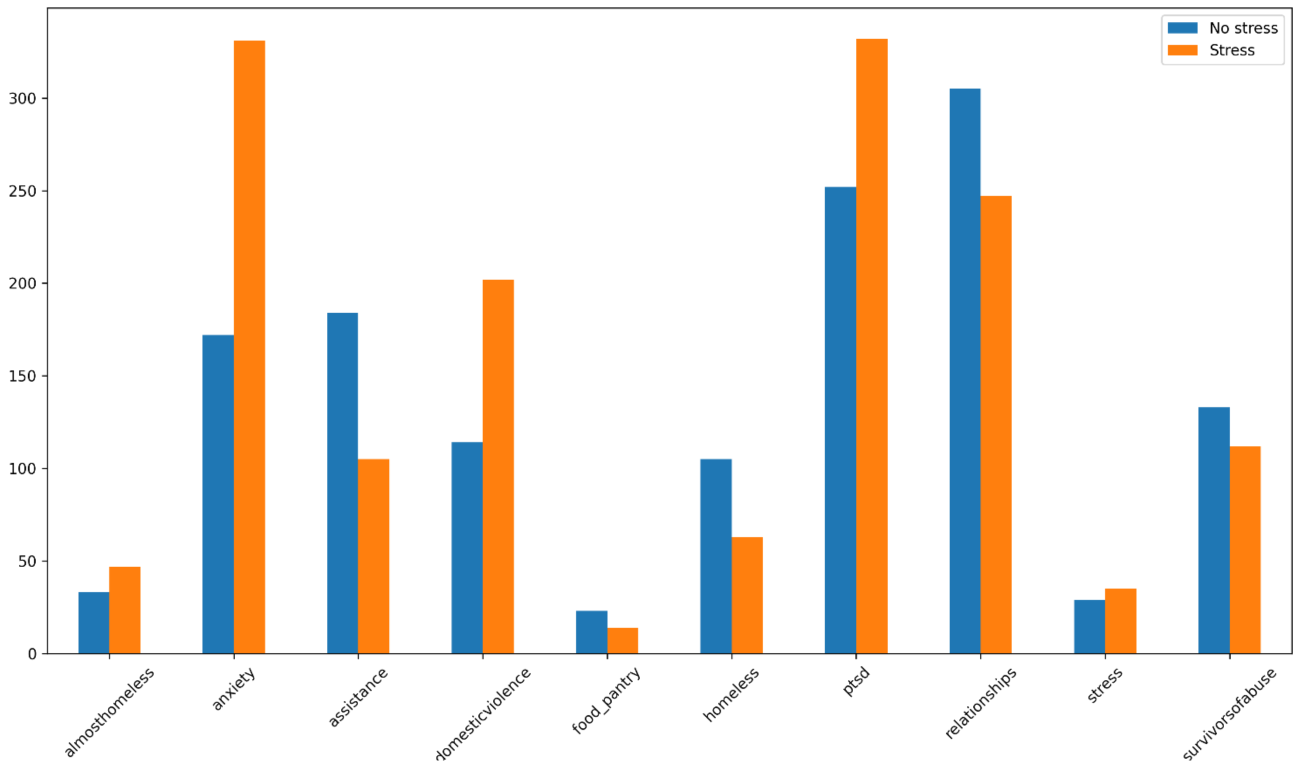


Figure 5. Subreddit – Stress label graph

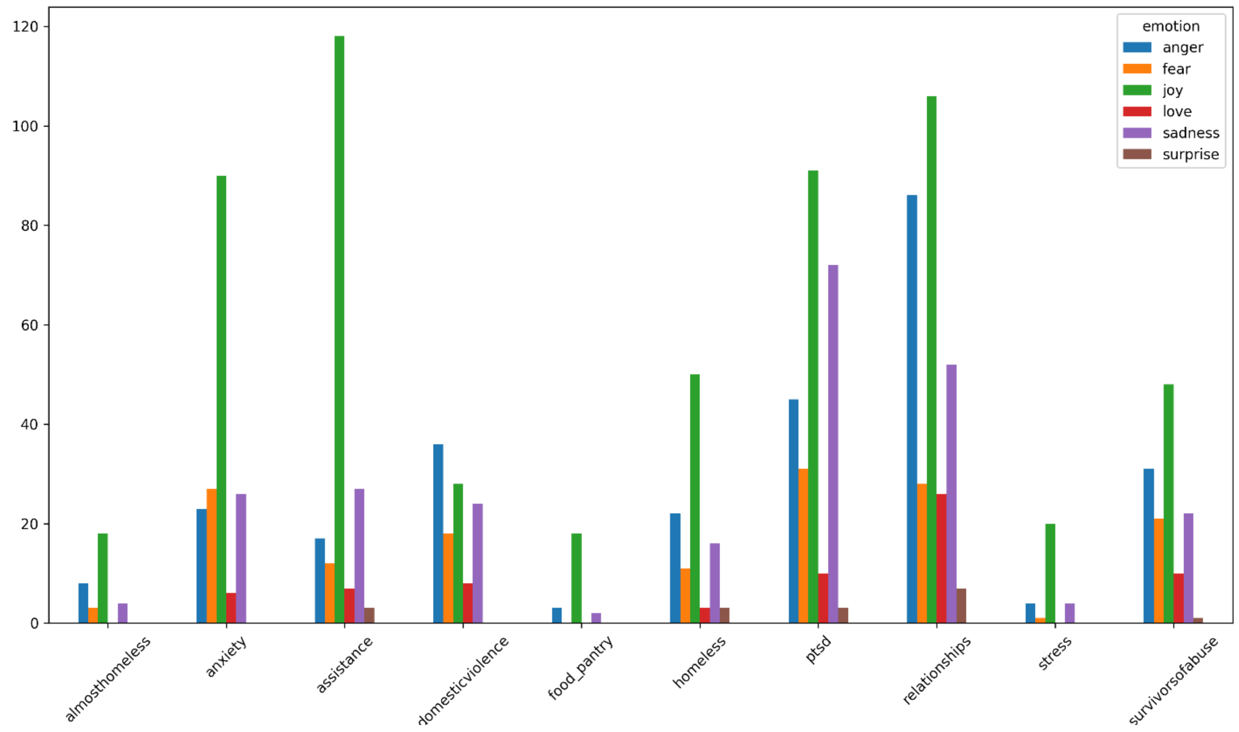


Figure 6. Label: **Not stressed** -> Subreddit – Emotion graph

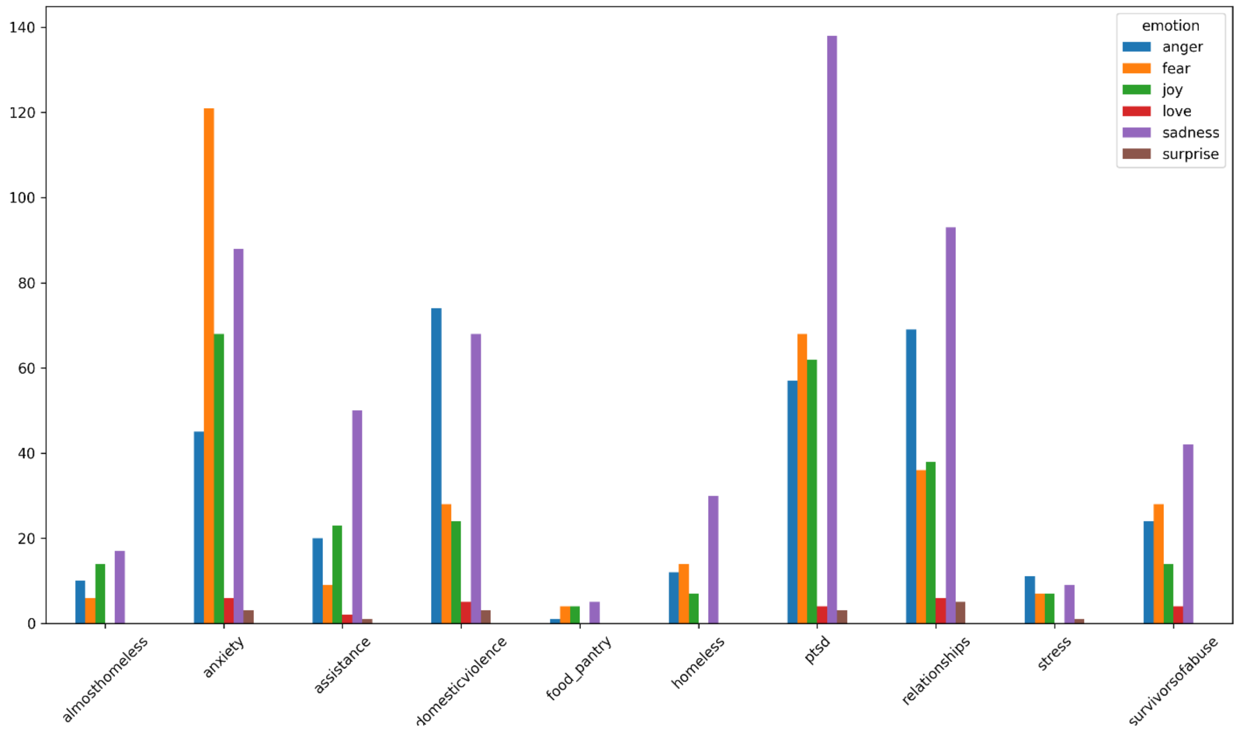


Figure 7. Label: **Stressed** -> Subreddit – Emotion graph

As it is shown in the figures, there is a relation between stress and emotion labels, preparing the confirmation of the hypothesis that emotions and emotive features in general, could have a great impact and optimize a mental health related classifier such ours (stress text classifier).

In the case of “no stress” the emotion of joy is the most frequent, whereas in the “stress” case the emotions that prevail are sadness, anger and fear (all the mainly negative emotions of the label set).

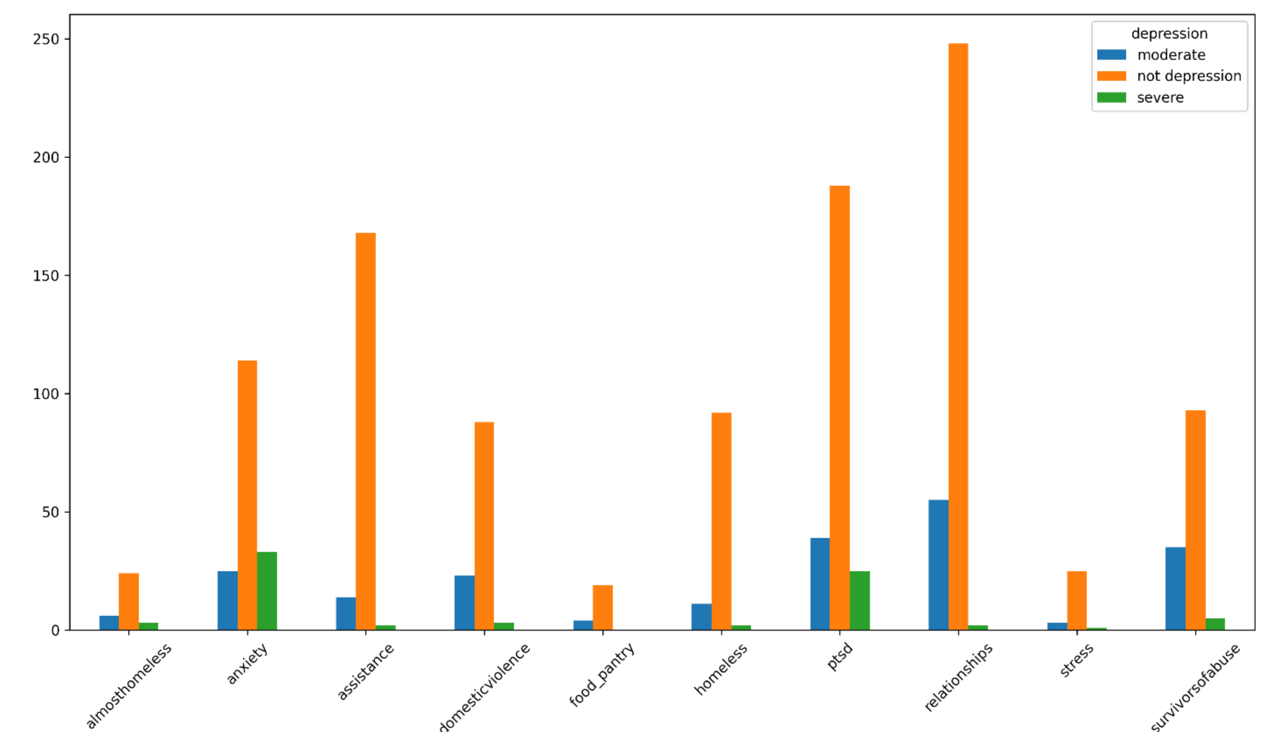


Figure 8. Label: **Not stressed** -> Subreddit – Depression graph

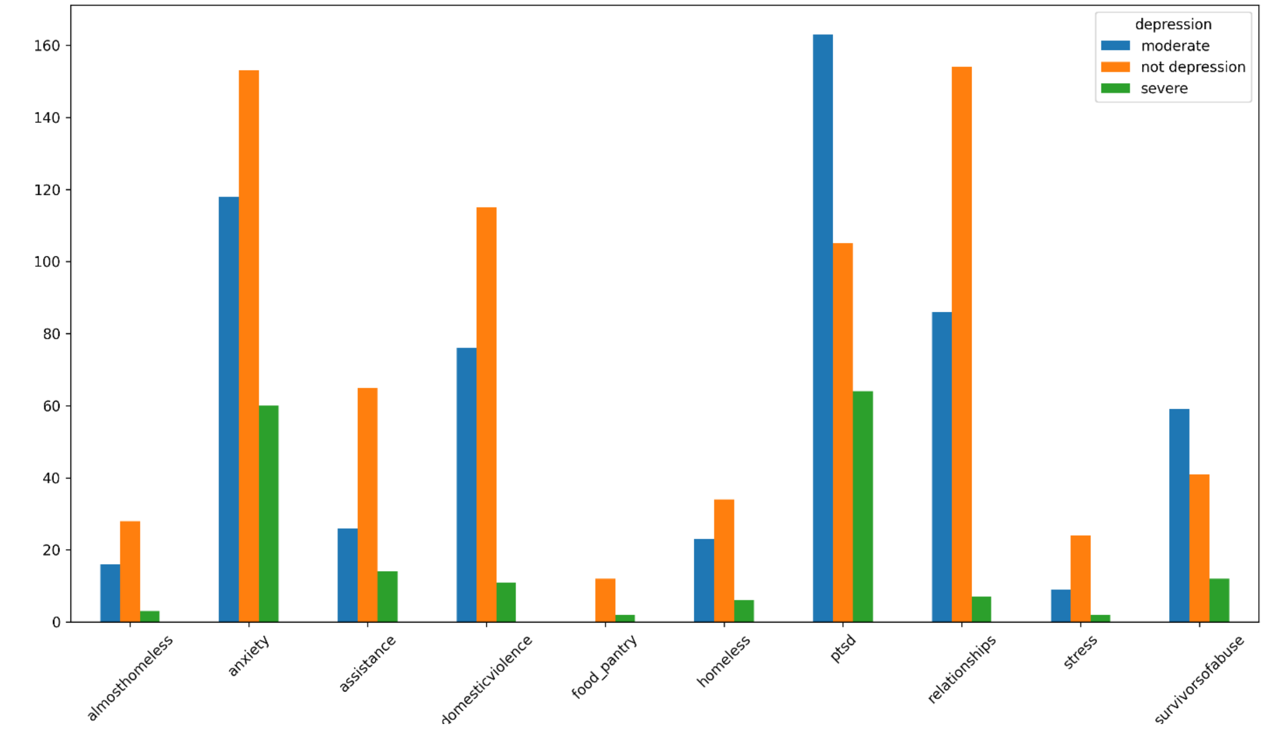


Figure 9. Label: **Stressed** -> Subreddit – Depression graph

The same applies to the depression labels. In “no stress” case, there are not so many cases of “moderate” or “severe” depression compared to the “stress” case, were they prevail.

Implementation

Methods

Classification algorithms

1. **Logistic Regression** (**C**: $5e^{-5}$, **solver**: lbfgs, **max_iter**: 1000, no data scaling)
2. **K Nearest Neighbors** (**n_neighbors**: 5 , no data scaling)
3. **Support Vector Machine** (**C**: 1.0, **kernel**: rbf, no data scaling)
4. **Random Forest** (**n_estimators**: 100, standard scaling)

Artificial Neural Networks

Layers	Model #1: **BiLSTM**	Model #2: **CNN**	Model #3: **BiGRU**
1	Dense (32 units, ReLU, L1-L2)	Convolutional (16 filters, size 2, ReLU)	Bidirectional GRU (32 units)
2	Bidirectional LSTM (32 units, 0.2 dropout, ReLU, L1-L2)	Max Pooling (size 2)	Dropout (0.2)
3	Dense (8 units, ReLU)	Convolutional (8 filters, size 2, ReLU)	Dense (64 units, ReLU)
4	Dropout (0.2)	Max Pooling (size 2)	Dropout (0.2)
5	Dense (1 unit, Sigmoid)	Flatten	Dense (32 units, ReLU)
6		Dense (8 units, ReLU)	Dropout (0.2)
7		Dropout (0.15)	Dense (1 unit, Sigmoid)
8		Dense (1 unit, Sigmoid)	

Notes

1. Lower Learning rate makes learning curve smoother
2. Overfitting can be confined by adding dropout layers or/and lower the unit number
3. Low batch size increases recall / lowers loss
4. In the case of BiLSTM, **standard scaler** works better than **normalization** or **regularization**
5. In the case of CNN, **normalization** works better than **standard scaler** or **regularization**

Results

Metrics

The models were evaluated using multiple metrics such as **Accuracy**, **Precision**, **Recall** and **F1** (harmonic Precision - Recall mean).

However, **recall** is more important in tasks related to health conditions like the stress detection, as it is better to diagnose someone not stressed as stressed than the contrary (fail in detecting it, which can be crucial !). Recall is the **proportion of actual positives was identified correctly (sensitivity)**. However, high sensitivity with extremely low specificity results in a useless classifier, labeling everything as positive.

A system **with high recall but low precision** returns many results, but most of its predicted labels are incorrect when compared to the training labels. A system with high precision but low recall is just the opposite, returning very few results, but most of its predicted labels are correct when compared to the training labels. An ideal system with high precision and high recall will return many results, with all results labeled correctly.

As a result, we try to balance the two metrics (precision & recall of the classes) with a slight focus and positive bias on recall as explained above. F1 score, indicates such balance.

Here, **macro averaging** is mostly used, as perhaps the most straightforward among the numerous averaging methods.

Learning Curve

When a trained model, the result might be not indicative of its performance and may depend on the test set. Therefore, learning curves are needed to monitor the overall training progress as well as spot unsolicited behaviour like overfitting or underfitting the training dataset, or even an unexpected result that might be owed to a coding mistakes and human error.

ROC Curves

This kind of curve displays the model's detection/sensitivity & "false alarm" probability at various thresholds. The greater the true positive rate and the lower the false positive rate, the better the class diagnostic ability of the model.

Best Model

In general, there are multiple ways to define a model "best", meaning there are various measures and ways to choose which suits you best depending on the task working on. For example, there might be a task where false positives are dangerous and others where false negatives are (such as in our case). Respectively, in the first case we need a high class precision in order to reduce the possibility of the false positives and in the second case high recall value so as to avoid the classification of positives as negatives as much as possible, therefore we focus on those metrics to choose the best model.

Logistic Regression	**Recall** (macro averaged)		**Precision** (macro averaged)		**F1** (macro averaged)	
	BERT	Word2Vec	BERT	Word2Vec	BERT	Word2Vec
AllFeats	.75	.73	.75	.73	.75	.73
Only Text Embeddings (no extra features)	.50	.50	.26	.26	.34	.34
Only Features (no text embeddings)	.75	.73	.75	.73	.75	.73
DAL	.50	.50	.26	.26	.34	.34
Emotion – Depression	.73	.71	.75	.73	.72	.71
Only Emotion Depression	.72	.71	.74	.73	.72	.71
Lexical	.74	.73	.75	.73	.74	.73
Only Lexical	.75	.73	.75	.73	.75	.73
LIWC	.74	.73	.75	.73	.74	.73
Only LIWC	.75	.73	.75	.73	.75	.73

k Nearest Neighbors	**Recall** (macro averaged)		**Precision** (macro averaged)		**F1** (macro averaged)	
	BERT	Word2Vec	BERT	Word2Vec	BERT	Word2Vec
AllFeats	.71	.82	.72	.83	.71	.82
Only Text Embeddings (no extra features)	.51	.50	.51	.49	.51	.48
Only Features (no text embeddings)	.71	.82	.72	.83	.71	.82
DAL	.53	.52	.53	.53	.53	.51
Emotion – Depression	.70	.77	.70	.77	.70	.77
Only Emotion Depression	.70	.77	.71	.77	.70	.77
Lexical	.70	.80	.71	.81	.69	.81
Only Lexical	.69	.80	.70	.81	.69	.81
LIWC	.70	.80	.71	.81	.69	.81
Only LIWC	.69	.80	.70	.81	.69	.81

Support Vector Machine	**Recall** (macro averaged)		**Precision** (macro averaged)		**F_1** (macro averaged)	
	BERT	Word2Vec	BERT	Word2Vec	BERT	Word2Vec
AllFeats	.76	.75	.77	.75	.76	.75
Only Text Embeddings (no extra features)	.49	.50	.44	.50	.36	.50
Only Features (no text embeddings)	.76	.75	.76	.75	.76	.75
DAL	.49	.58	.49	.59	.41	.58
Emotion – Depression	.74	.74	.74	.74	.74	.74
Only Emotion Depression	.74	.73	.74	.74	.74	.73
Lexical	.75	.74	.76	.75	.75	.74
Only Lexical	.75	.74	.75	.75	.75	.74
LIWC	.75	.75	.76	.75	.75	.74
Only LIWC	.75	.75	.75	.75	.75	.75

Random Forest	**Recall** (macro averaged)		**Precision** (macro averaged)		**F_1** (macro averaged)	
	BERT	Word2Vec	BERT	Word2Vec	BERT	Word2Vec
AllFeats	.73	.70	.73	.71	.73	.70
Only Text Embeddings (no extra features)	.52	.50	.52	.50	.52	.50
Only Features (no text embeddings)	.75	.77	.75	.73	.75	.75
DAL	.52	.50	.52	.50	.52	.50
Emotion – Depression	.61	.59	.61	.59	.61	.59
Only Emotion Depression	.74	.77	.75	.70	.74	.74
Lexical	.71	.67	.71	.67	.71	.67
Only Lexical	.75	.75	.75	.73	.75	.73
LIWC	.71	.67	.72	.67	.71	.67
Only LIWC	.75	.76	.75	.73	.75	.74

Bi LSTM	**Recall** (macro averaged)		**Precision** (macro averaged)		**F_1** (macro averaged)	
	BERT	Word2Vec	BERT	Word2Vec	BERT	Word2Vec
AllFeats	.74	.74	.76	.74	.74	.74
Only Text Embeddings (no extra features)	.50	.50	.26	.26	.34	.34
Only Features (no text embeddings)	.74	.73	.74	.73	.74	.73
DAL	.50	.50	.26	.26	.34	.34
Emotion – Depression	.50	.71	.26	.71	.34	.71
Only Emotion Depression	.73	.75	.74	.68	.73	.66
Lexical	.73	.72	.73	.73	.73	.72
Only Lexical	.73	.73	.75	.74	.73	.73
LIWC	.74	.72	.74	.72	.74	.72
Only LIWC	.73	.71	.74	.73	.72	.70

CNN	**Recall** (macro averaged)		**Precision** (macro averaged)		**F_1** (macro averaged)	
	BERT	Word2Vec	BERT	Word2Vec	BERT	Word2Vec
AllFeats	.72	.72	.72	.72	.72	.72
Only Text Embeddings (no extra features)	.50	.52	.44	.53	.35	.46
Only Features (no text embeddings)	.71	.67	.72	.67	.71	.67
DAL	.50	.50	.50	.48	.38	.35
Emotion – Depression	.69	.72	.71	.73	.69	.72
Only Emotion Depression	.67	.70	.69	.72	.66	.70
Lexical	.73	.71	.73	.71	.73	.71
Only Lexical	.65	.69	.65	.70	.65	.69
LIWC	.72	.67	.73	.67	.72	.67
Only LIWC	.69	.68	.70	.70	.69	.68

Bi GRU	Recall (macro averaged)		Precision (macro averaged)		F_1 (macro averaged)	
	BERT	Word2Vec	BERT	Word2Vec	BERT	Word2Vec
AllFeats	.73	.75	.74	.76	.74	.75
Only Text Embeddings (no extra features)	.50	.50	.50	.50	.42	.50
Only Features (no text embeddings)	.72	.73	.72	.73	.72	.73
DAL	.57	.51	.46	.51	.43	.50
Emotion – Depression	.74	.71	.74	.72	.74	.71
Only Emotion Depression	.72	.72	.73	.73	.72	.72
Lexical	.75	.74	.76	.74	.75	.74
Only Lexical	.73	.74	.73	.74	.73	.74
LIWC	.76	.73	.76	.74	.76	.73
Only LIWC	.73	.73	.73	.74	.73	.73

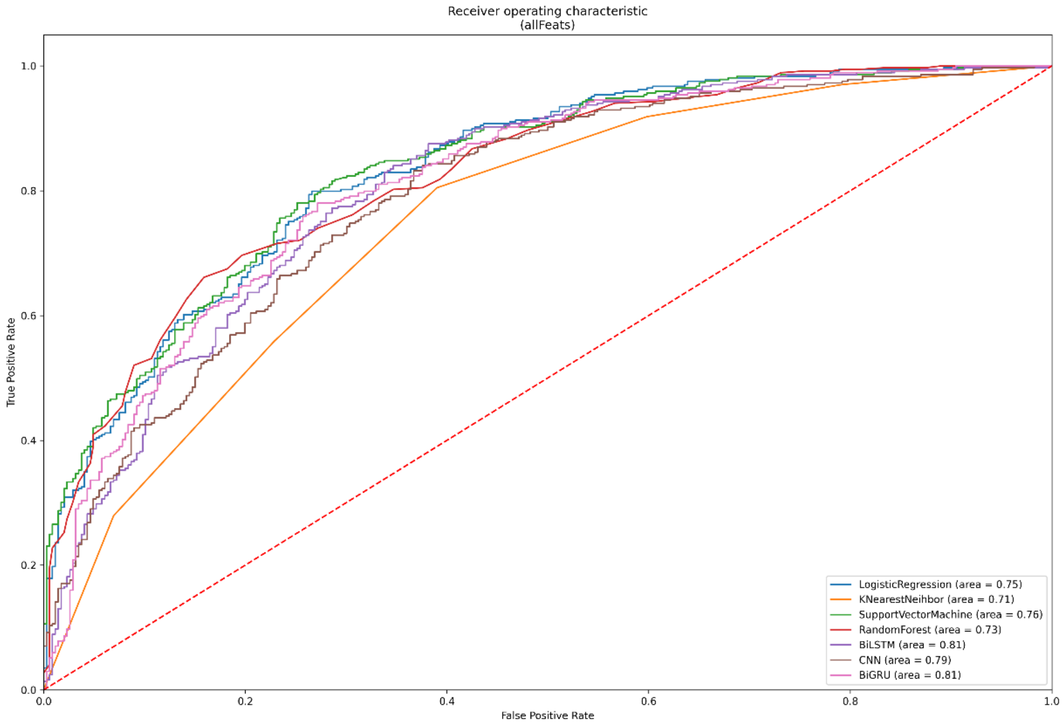
It seems that in some cases, the features extracted from the text casted the text itself redundant. More importantly, the additional embedding features might not only be of no use for the models, but probably increase the complexity, creating “obstacles” for the models during training.

Comparison

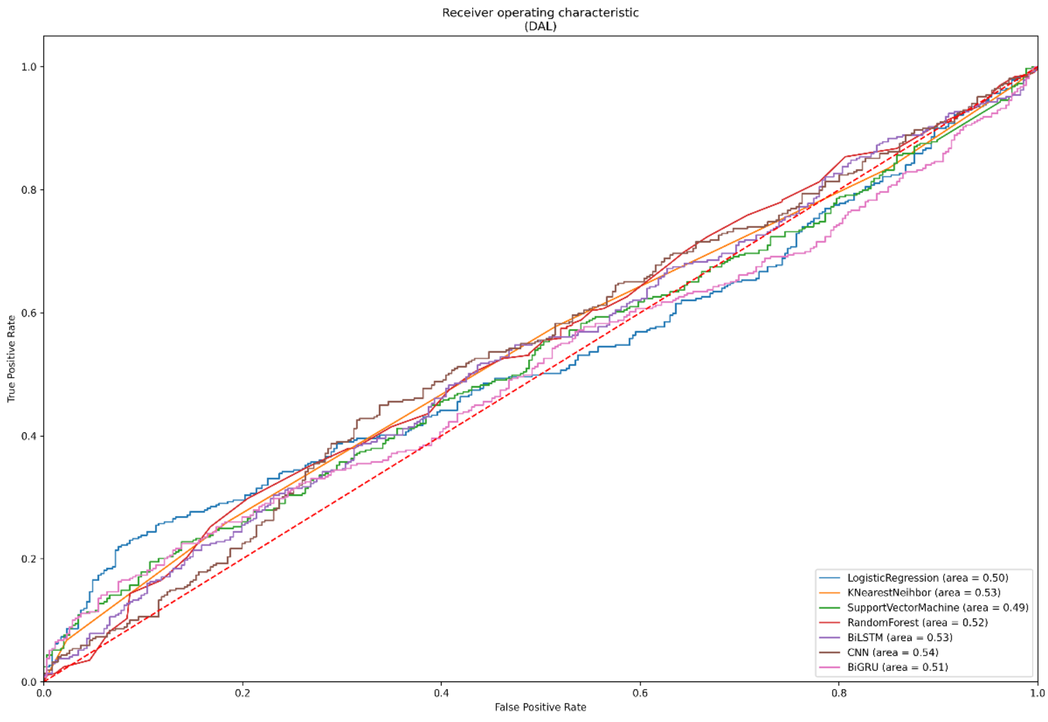
Below, the **ROC Curve** and **Loss / Accuracy / Precision / Recall Curves** of remarkable cases are shown for comparison as well as pattern or conclusion extraction.
(More figures are included in the .ppt presentation of the project.)

ROC Curves

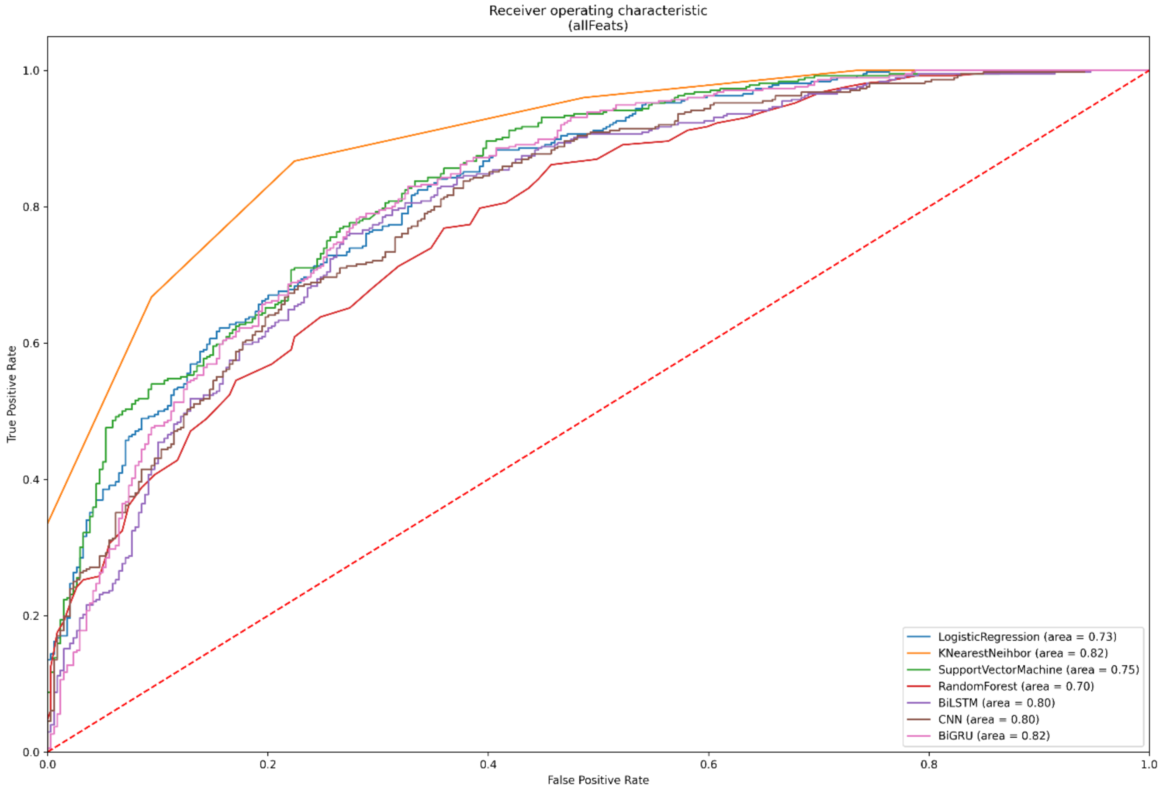
BERT – All Features



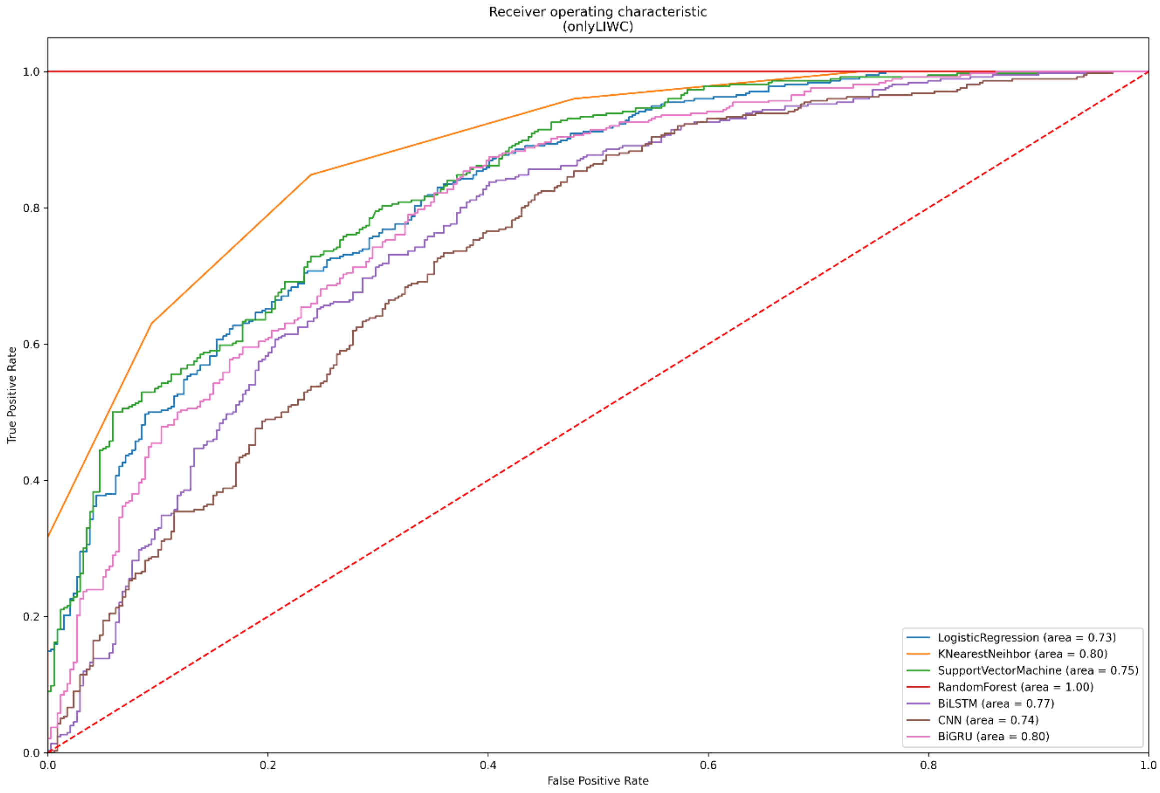
BERT – DAL Features



Word2Vec – All Features

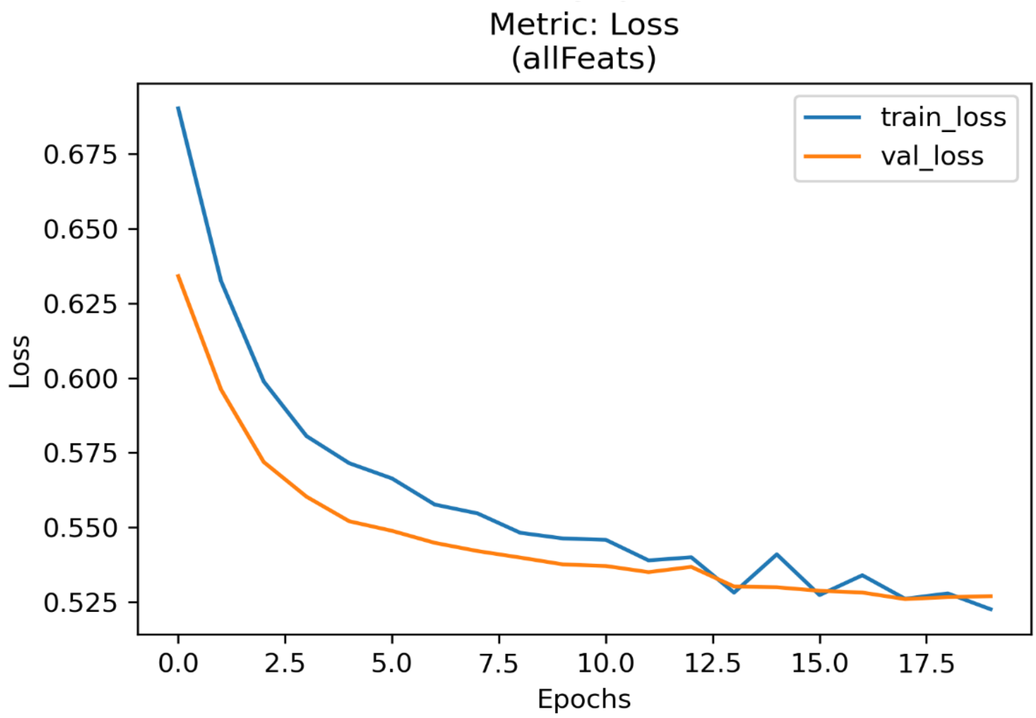


Word2Vec – only LIWC Features

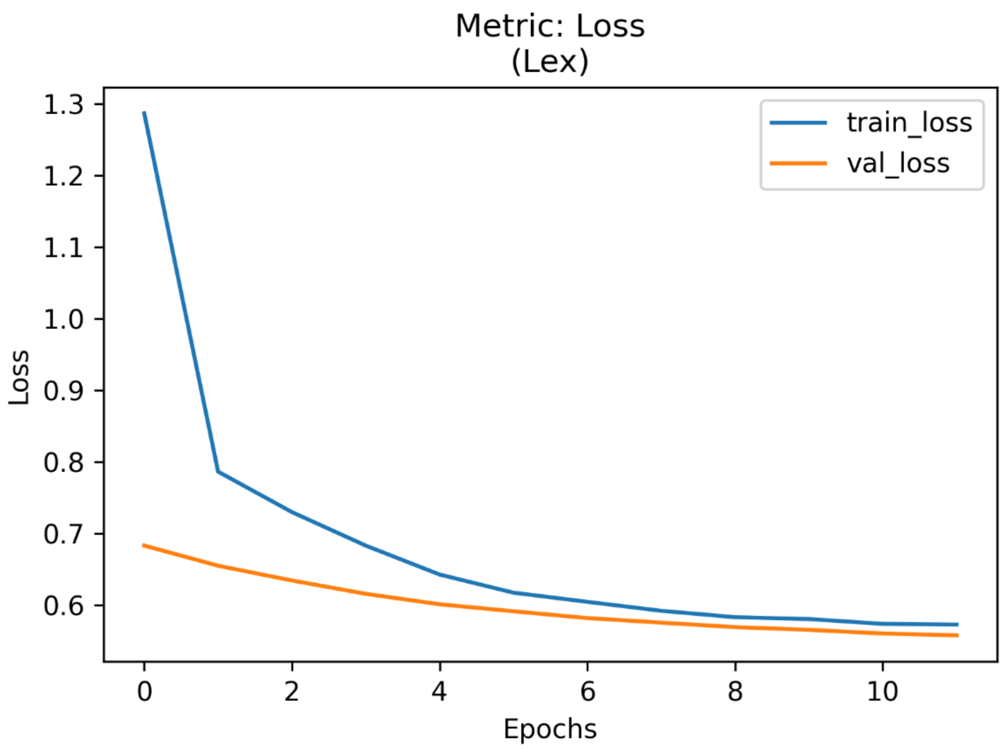


Learning Curves

BERT – All Features - BiGRU



Word2Vec – Lexical Features - CNN



Conclusions

Multiple conclusions could be drawn from the results, the figures, the performance scores and the log files of the models (stored in the .zip file of the project as mentioned above).

To begin with, stress detection comprises a medical task, crucial for the **mental health** of numerous people of all ages, therefore the ability of a classifier to detect stress where there is stress, is more important than the general ability to predict correctly (high sensitivity/**recall**, high f1 score).

Furthermore, it seems that **KNN** (K Nearest Neighbors) classification algorithm and **BiGRU** (Bidirectional GRU Gated Recurrent Unit) outperformed the rest of the models. More specifically, KNN was able to handle the total set of features (Word2Vec embeddings) better than the rest, and between the NNs BiGRU performed much better in total (class/micro averaged recall, precision, F_1). In different feature combinations, BiGRU seems to also perform consistently well compared to its competitors, even with such a small training set.

Neural networks seemed to have a tendency for quick overfitting due to lack of data samples compared to non NNs.

An important conclusion is that emotion features actually seem to improve classifier sensitivity, confirming one of our initial hypotheses on emotion related features.

Finally, stress detection proves to be a lexical problem, therefore this type of attributes (lexical e.g. LIWC, DAL, etc.) could be used as an asset in the development of resources with psychological application development.

Future Work

The project at hand is called “mini”, due to the fact that it was implemented only in a few days of hard work and dedication. However, during the process multiple new goals were set and questions were risen. As a result, the following key points could be the keywords/notes of a new, sequential project, as an optimization and further development of the present:

1. Insufficient data create the need to further work for dataset **enrichment** with more useful features
2. More **Experiments** could be performed:
   a. Model types
   b. Model fine-tuning
   (grid search – time consuming, fine-tune on several feature combinations)
   c. Features
   d. Text embeddings (e.g. GloVe)
   e. Subreddit categories / threads
   f. Further data scaling

References

(Articles & Papers the project at hand was based on)

1. Dreaddit: A Reddit Dataset for Stress Analysis in Social Media
2. The Development and Psychometric Properties of LIWC2015
3. Pattern Library for Natural Language Processing
4. The Whissell Dictionary of Affect in Language
5. Automated readability index
6. Flesch-Kincaid Grade Level
7. Distilbert-base-uncased-emotion
8. Keras vs Tensorflow vs Pytorch: Key Differences Among the Deep Learning Framework
9. Machine Learning Classifiers - The Algorithms & How They Work
10. Multi-Class Classification Tutorial with the Keras Deep Learning Library
11. Enriching BERT with Knowledge Graph Embeddings for Document Classification
12. BERT Word Embeddings Tutorial

Links

1. Text Classification | Kaggle
2. 1911.00133v1.pdf
3. Shared with me - Google Drive
4. M908_miniProject | Kaggle
5. BERT get sentence embedding – Python
6. Regularization Techniques And Their Implementation In TensorFlow(Keras) | by Richmond Alake | Towards Data Science
7. BERT Word Embeddings Tutorial · Chris McCormick
8. BERT Sentence Embeddings.ipynb
9. 2011.05864.pdf
10. NLP Essential Guide: Convolutional Neural Network for Sentence Classification | cnvrg.io
11. Micro, Macro & Weighted Averages of F1 Score, Clearly Explained | by Kenneth Leung | Towards Data Science
12. Keras - Model Evaluation and Model Prediction
13. Stress Detection with Machine Learning
14. Tensorflow vs PyTorch for Text Classification using GRU | by Rodolfo Saldanha | The Startup | Medium
15. Text Classification - Deep Learning CNN Models.ipynb - Colaboratory
16. [1408.5882] Convolutional Neural Networks for Sentence Classification
17. Precision vs Recall | Precision and Recall Machine Learning
18. Data Science in Medicine — Precision & Recall or Specificity & Sensitivity? | by Alon Lekhtman | Towards Data Science
19. NLP: Contextualized word embeddings from BERT | by Andreas Pogiatzis | Towards Data Science
20. A Comprehensive Guide to Understand and Implement Text Classification in Python
21. How I improved my text classification model with feature engineering | by Alexandre Wrg | Towards Data Science
22. Whissell Dictionary of Affect in Language - Freeware
23. Pattern Library for NLP in Python - Analytics Vidhya
24. LIWC2015_LanguageManual.pdf
25. NLP: Word2Vec with Python Example | by Amit Chauhan | The Pythoneers | Medium
26. Basics of Using Pre-trained GloVe Vectors in Python | by Sebastian Theiler | Analytics Vidhya | Medium
27. FastText Python - Learn Word Representations
28. pip suppress warning root - Google Search
29. models.word2vec – Word2vec embeddings — gensim
30. RaRe-Technologies/gensim-data: Data repository for pretrained NLP models and NLP corpora.
31. word 2 vec encode - Google Search
32. BERT tokenizer with 9 models-NLP stress analysis | Kaggle
33. 1911.00133v1.pdf
34. Project4__25per100 - Google Drive
35. M908_miniProject | Kaggle
36. BERT get sentence embedding – Python
37. Regularization Techniques And Their Implementation In TensorFlow(Keras) | by Richmond Alake | Towards Data Science
38. BERT Word Embeddings Tutorial · Chris McCormick
39. BERT Sentence Embeddings.ipynb
40. 2011.05864.pdf
41. NLP Essential Guide: Convolutional Neural Network for Sentence Classification | cnvrg.io
42. Micro, Macro & Weighted Averages of F1 Score, Clearly Explained | by Kenneth Leung | Towards Data Science
43. Keras - Model Evaluation and Model Prediction
44. Stress Detection with Machine Learning
45. Tensorflow vs PyTorch for Text Classification using GRU | by Rodolfo Saldanha | The Startup | Medium
46. Text Classification - Deep Learning CNN Models.ipynb - Colaboratory
47. [1408.5882] Convolutional Neural Networks for Sentence Classification
48. Text classification with an RNN | TensorFlow
49. Multi-Class Classification Tutorial with the Keras Deep Learning Library
50. Enriching BERT with Knowledge Graph Embeddings for Document Classification - 1909.08402.pdf
51. Text Classification using Neural Networks | Kaggle
52. Deep Learning Techniques for Text Classification | by Diardano Raihan | Towards Data Science
53. diardanoraihan/Text_Classification_Capstone: Evaluate the performance of several state-of-the-art deep learning techniques on various text classification datasets. This project is part of Master's Thesis in Computer Control and Automation, NTU.
54. Text Classification with Neural Networks - Atmosera
55. Practical Text Classification With Python and Keras – Real Python
56. How to solve Multi-Class Classification Problems in Deep Learning with Tensorflow & Keras? | by Murat Karakaya | Deep Learning Tutorials with Keras | Medium
57. Bidirectional LSTM on IMDB
58. Embedding layer
59. add metadata to text embedding bert - Google Search
60. Incorporating Metadata into Content-Based User Embeddings - W17-4406.pdf
61. Layer weight regularizers
62. The Sequential model
63. sklearn.model_selection.cross_val_score — scikit-learn 1.1.1 documentation
64. python - Scikit-learn TypeError: If no scoring is specified, the estimator passed should have a 'score' method - Stack Overflow
65. The first argument to `Layer.call` must always be passed. - Google Search
66. How to Choose Loss Functions When Training Deep Learning Neural Networks
67. A Deep Learning Model to Perform Keras Binary Classification | Pluralsight
68. Optimizers
69. Accuracy metrics
70. python - Keras model not training layers, validation accuracy always 0.5 - Stack Overflow
71. python - Keras deep learning why validation accuracy stuck in a value every time? - Stack Overflow
72. New Tab
73. Metrics
74. machine learning - How to get accuracy, F1, precision and recall, for a keras model? - Data Science Stack Exchange
75. tf.keras.Model | TensorFlow Core v2.9.1
76. Implementation of SimpleRNN, GRU, and LSTM Models in Keras and Tensorflow For an NLP Project – Regenerative
77. How to add function (Get F1-score) in Keras metrics and record F1 value after each epoch? | by Aakash Goel | Medium
78. 05_Pre_Trained_Word_Embeddings.ipynb - Colaboratory
79. NLP: Contextualized word embeddings from BERT | by Andreas Pogiatzis | Towards Data Science
80. BERT Word Embeddings Tutorial · Chris McCormick
81. Frontiers | Using Neural Networks to Generate Inferential Roles for Natural Language | Psychology
82. Natural Language Analysis and the Psychology of Verbal Behavior: The Past, Present, and Future States of the Field - Ryan L. Boyd, H. Andrew Schwartz, 2021
83. [1202.0116] Inference and Plausible Reasoning in a Natural Language Understanding System Based on Object-Oriented Semantics
84. Sentiment and Emotion Analysis API
85. NLP applications: emotion analysis and natural language inference
86. emotion-survey.pdf
87. Emotion Action Detection and Emotion Inference: the Task and Dataset – arXiv Vanity
88. 1902.00679.pdf
89. emotional inference nlp - Google Scholar
90. emotion causality emotion inference - Google Search
91. Natural language processing in mental health applications using non-clinical texts† | Natural Language Engineering | Cambridge Core
92. Enhancing emotion inference in conversations with commonsense knowledge - ScienceDirect
93. Comparative Analyses of Bert, Roberta, Distilbert, and Xlnet for Text-Based Emotion Recognition | IEEE Conference Publication | IEEE Xplore
94. Transformer models for text-based emotion detection: a review of BERT-based approaches | SpringerLink
95. Emotion Analysis - an overview | ScienceDirect Topics
96. (PDF) A Puzzle-Based Dataset for Natural Language Inference
97. Natural Logic and Natural Language Inference | Request PDF
98. (PDF) Enhanced LSTM for Natural Language Inference
99. (PDF) Generating Natural Language Inference Chains
100. (PDF) Identifying inherent disagreement in natural language inference
101. (PDF) NeuralLog: Natural Language Inference with Joint Neural and Logical Reasoning
102. (PDF) NeuralLog: Natural Language Inference with Joint Neural and Logical Reasoning
103. SciNLI: A Corpus for Natural Language Inference on Scientific Text | Request PDF
104. (PDF) Deep Learning for Natural Language Inference
105. Deep Learning for Natural Language Inference

106. (PDF) Natural language inference by deep learning method
107. Uncertain Natural Language Inference | Request PDF
108. Learning Natural Language Inference with LSTM | Request PDF
109. Dialogue Natural Language Inference
110. (PDF) Natural language inference for clinical registry curation
111. (PDF) NILE : Natural Language Inference with Faithful Natural Language Explanations
112. (PDF) Semantic Diversity in Dialogue with Natural Language Inference
113. Temporal Reasoning in Natural Language Inference | Request PDF
114. text inference emotion nlp - Google Scholar
115. Recurrent synchronization network for emotion-cause pair extraction - ScienceDirect
116. [2105.05541] Evaluating Gender Bias in Natural Language Inference
117. [2203.06728] SciNLI: A Corpus for Natural Language Inference on Scientific Text
118. Deep learning for affective computing: Text-based emotion recognition in decision support - ScienceDirect
119. Fine-Grained Emotion Strength Transfer, Control and Prediction for Emotional Speech Synthesis | IEEE Conference Publication | IEEE Xplore
120. Mining Social Emotions from Affective Text | IEEE Journals & Magazine | IEEE Xplore
121. Using YouTube Comments for Text-based Emotion Recognition - ScienceDirect
122. Automatic Human Emotion Classification in Web Document Using Fuzzy Inference System (FIS): Human Emotion Classification: Social Sciences & Humanities Journal Article | IGI Global
123. Frontiers | Simulating Emotions: An Active Inference Model of Emotional State Inference and Emotion Concept Learning | Psychology
124. User group based emotion detection and topic discovery over short text | SpringerLink
125. Emotion-enhanced classification based on fuzzy reasoning | SpringerLink
126. Computational Models of Emotion Inference in Theory of Mind: A Review and Roadmap - Ong - 2019 - Topics in Cognitive Science - Wiley Online Library
127. Sensors | Free Full-Text | Emotion Detection for Social Robots Based on NLP Transformers and an Emotion Ontology
128. Detecting implicit expressions of emotion in text: A comparative analysis - ScienceDirect
129. Identifying Emotions in Social Media: Comparison of Word-Emotion Lexicons | IEEE Conference Publication | IEEE Xplore
130. Emotion Detection with Natural Language Inference | Open Data Science Conference
131. Stress Analysis in Social Media | Kaggle
132. M908_miniProject | Kaggle
133. Dreaddit: A Reddit Dataset for Stress Analysis in Social Media - D19-6213.pdf
134. 1911.00133v1.pdf
135. LIWC — How It Works
136. Advanced_Data_Visualization
137. BERT tokenizer with 9 models-NLP stress analysis | Kaggle
138. LIWC2015_LanguageManual.pdf
139. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank - D13-1170.pdf
140. desmedt12a.dvi - desmedt12a.pdf
141. Pattern Library for NLP in Python - Analytics Vidhya
142. Using the Revised Dictionary of Affect in Language to quantify the emotional undertones of samples of natural language - PubMed
143. Whissell Dictionary of Affect in Language - Freeware
144. pandas.DataFrame.plot.scatter — pandas 0.25.0 documentation
145. TausczikPennebaker2010.pdf
146. Flesch Reading Ease and the Flesch Kincaid Grade Level – Readable
147. bhadresh-savani/distilbert-base-uncased-emotion · Hugging Face
148. Pandas Scatter Plot: How to Make a Scatter Plot in Pandas • datagy
149. huggingface/transformers: 🤗 Transformers: State-of-the-art Machine Learning for Pytorch, TensorFlow, and JAX.
150. 68558.pdf
151. Microsoft Word - 1985072021122740Arushi IEEE COG 2021_v12.doc - paper_284.pdf
152. BERT
153. Fine-tune a pretrained model
154. Hugging Face – The AI community building the future.
155. [1408.5882] Convolutional Neural Networks for Sentence Classification
156. 1809.04505.pdf
157. pxuab/emo2vec_wassa_paper
158. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding - N19-1423.pdf
159. diardanoraihan/Text_Classification_Capstone: Evaluate the performance of several state-of-the-art deep learning techniques on various text classification datasets. This project is part of Master's Thesis in Computer Control and Automation, NTU.
160. Text Classification with Neural Networks - Atmosera
161. Practical Text Classification With Python and Keras – Real Python
162. Text classification with an RNN | TensorFlow
163. Combining numerical and text features in deep neural networks | by Christian Freischlag | Towards Data Science
164. Keras vs Tensorflow vs Pytorch [Updated] | Deep Learning Frameworks | Simplilearn
165. [Deep Learning] Combining numerical and text features in (deep) neural networks - Digital Thinking
166. Machine Learning Classifiers - The Algorithms & How They Work
167. Multi-Class Classification Tutorial with the Keras Deep Learning Library
168. Bidirectional LSTM on IMDB
169. How to include metadata information on sentence embeddings? : LanguageTechnology
170. Incorporating Metadata into Content-Based User Embeddings - W17-4406.pdf
171. Enriching BERT with Knowledge Graph Embeddings for Document Classification - 1909.08402.pdf
172. Text classification with an RNN | TensorFlow
173. BERT Word Embeddings Tutorial · Chris McCormick
174. Google Colab
175. bert-for-inference/introduction-to-bert.ipynb at master · BramVanroy/bert-for-inference
176. Smart Batching Tutorial - Speed Up BERT Training · Chris McCormick
177. Smart Batching Tutorial - Speed Up BERT Training · Chris McCormick
178. M931_Ai2_Project4_CTKylafi_LT1200012.ipynb - Colaboratory
179. bert-for-inference/introduction-to-bert.ipynb at master · BramVanroy/bert-for-inference
180. sentence-transformers/bert-base-nli-cls-token · Hugging Face
181. BERT Word Embeddings Tutorial · Chris McCormick
182. BERT
183. Captum · Model Interpretability for PyTorch
184. Smart Batching Tutorial - Speed Up BERT Training · Chris McCormick
185. How to turn a list of tensor to tensor? - PyTorch Forums
186. Random sample of paired lists in Python - Stack Overflow
187. Accuracy, Precision, Recall & F1-Score - Python Examples - Data Analytics
188. Regularization Techniques And Their Implementation In TensorFlow(Keras) | by Richmond Alake | Towards Data Science
189. RMSprop
190. acc and val_acc don't change? · Issue #1597 · keras-team/keras
191. 4 Sentence Embedding Techniques One Should Know| With Python Codes
192. BERT get sentence embedding – Python
193. BERT Word Embeddings v2.ipynb - Colaboratory
194. Sentence level embeddings from BERT | DAIR.AI
195. BERT Sentence Embeddings.ipynb
196. BERT Word Embeddings Deep Dive. Dives into BERT word embeddings with… | by Isanka Rajapaksha | Medium
197. BERT Word Embeddings Tutorial · Chris McCormick
198. NLP: Contextualized word embeddings from BERT | by Andreas Pogiatzis | Towards Data Science
199. bohanli/BERT-flow: TensorFlow implementation of On the Sentence Embeddings from Pre-trained Language Models (EMNLP 2020)
200. Understanding BERT — Word Embeddings | by Dharti Dhami | Medium
201. Hidden states embedding tensors - 🤗Transformers - Hugging Face Forums
202. 2011.05864.pdf
203. Micro, Macro & Weighted Averages of F1 Score, Clearly Explained | by Kenneth Leung | Towards Data Science
204. NLP Essential Guide: Convolutional Neural Network for Sentence Classification | cnvrg.io
205. ShawnyXiao/TextClassification-Keras: Text classification models implemented in Keras, including: FastText, TextCNN, TextRNN, TextBiRNN, TextAttBiRNN, HAN, RCNN, RCNNVariant, etc.
206. Convolutional Neural Networks for Sentence Classification | Papers With Code
207. Stress Detection with Machine Learning
208. Micro, Macro & Weighted Averages of F1 Score, Clearly Explained | by Kenneth Leung | Towards Data Science
209. Keras - Model Evaluation and Model Prediction
210. Tensorflow vs PyTorch for Text Classification using GRU | by Rodolfo Saldanha | The Startup | Medium
211. 1312.4400.pdf

212. Tensorflow vs PyTorch for Text Classification using GRU | by Rodolfo Saldanha | The Startup | Medium
213. TensorBoard
214. Text Classification - Deep Learning CNN Models.ipynb - Colaboratory
215. [1408.5882] Convolutional Neural Networks for Sentence Classification
216. facebookresearch/pytext: A natural language modeling framework based on PyTorch
217. yoonkim/CNN_sentence: CNNs for sentence classification
218. yoonkim/CNN_sentence: CNNs for sentence classification
219. SebastienPavot/Text-Classification-with-Keras: Python / Keras / Streamlit - Classify tweets using Keras models and present results in a Streamlit app.
220. Precision vs Recall | Precision and Recall Machine Learning
221. Precision vs Recall | Precision and Recall Machine Learning
222. Building A Logistic Regression in Python, Step by Step | by Susan Li | Towards Data Science
223. Precision-Recall — scikit-learn 1.1.1 documentation
224. Building A Logistic Regression in Python, Step by Step | by Susan Li | Towards Data Science
225. Text Classification - Deep Learning Sequential Models - Bidirectional GRUs with Attention.ipynb - Colaboratory
226. Simple guide on how to generate ROC plot for Keras classifier | DLology
227. Text Classification - Deep Learning Sequential Models - Bidirectional GRUs with Attention.ipynb - Colaboratory
228. How to Use StandardScaler and MinMaxScaler Transforms in Python
229. Classification: Precision and Recall | Machine Learning Crash Course | Google Developers
230. Simple guide on how to generate ROC plot for Keras classifier | DLology
231. Data Science in Medicine — Precision & Recall or Specificity & Sensitivity? | by Alon Lekhtman | Towards Data Science
232. BERT get sentence embedding – Python
233. Data Science in Medicine — Precision & Recall or Specificity & Sensitivity? | by Alon Lekhtman | Towards Data Science
234. Precision vs Recall | Precision and Recall Machine Learning
235. [1408.5882] Convolutional Neural Networks for Sentence Classification
236. Text Classification - Deep Learning CNN Models.ipynb - Colaboratory
237. Tensorflow vs PyTorch for Text Classification using GRU | by Rodolfo Saldanha | The Startup | Medium
238. Stress Detection with Machine Learning
239. Keras - Model Evaluation and Model Prediction
240. Micro, Macro & Weighted Averages of F1 Score, Clearly Explained | by Kenneth Leung | Towards Data Science
241. NLP Essential Guide: Convolutional Neural Network for Sentence Classification | cnvrg.io
242. 2011.05864.pdf
243. BERT Sentence Embeddings.ipynb
244. Regularization Techniques And Their Implementation In TensorFlow(Keras) | by Richmond Alake | Towards Data Science
245. BERT Word Embeddings Tutorial · Chris McCormick
246. A Comprehensive Guide to Understand and Implement Text Classification in Python
247. Text Classification | Kaggle
248. NLP: Contextualized word embeddings from BERT | by Andreas Pogiatzis | Towards Data Science
249. NLP: Word2Vec with Python Example | by Amit Chauhan | The Pythoneers | Medium
250. Basics of Using Pre-trained GloVe Vectors in Python | by Sebastian Theiler | Analytics Vidhya | Medium
251. FastText Python - Learn Word Representations
252. pip suppress warning root - Google Search
253. models.word2vec – Word2vec embeddings — gensim
254. RaRe-Technologies/gensim-data: Data repository for pretrained NLP models and NLP corpora.
255. word 2 vec encode - Google Search
256. RaRe-Technologies/gensim-data: Data repository for pretrained NLP models and NLP corpora.
257. Text Classification With Word2Vec - DS lore
258. models.keyedvectors – Store and query word vectors — gensim
259. awesome-sentence-embedding | A curated list of pretrained sentence and word embedding models
260. IBM/WordMoversEmbeddings: WordMoversEmbeddings(WME) is a simple code for generating the vector representation of sentence/document for text classification and clustering.
261. How to Get Vector for A Sentence From Word2vec of Tokens | Baeldung on Computer Science
262. machine learning - How to train sentence/paragraph/document embeddings? - Cross Validated
263. Word2Vec For Phrases — Learning Embeddings For More Than One Word | by Moshe Hazoom | Towards Data Science
264. RaRe-Technologies/gensim-data: Data repository for pretrained NLP models and NLP corpora.
265. shabeelkandi/Handling-Out-of-Vocabulary-Words-in-Natural-Language-Processing-using-Language-Modelling
266. https://cnvrg.io/cnn-sentence-classification/