Universität des Saarlandes
Prof. Dr. Dietrich Klakow
Lehrstuhl für Signalverarbeitung
NNTI Winter Term 2020/2021

Course Project

Sentiment Analysis & Transfer Learning

Deadline: 31.03.2021, 23:59

Introduction

This is a final project for the course *Neural Networks: Theory and Implementation (NNTI)*. This project will introduce you to Sentiment Classification and Analysis. *Sentiment analysis* (also known as opinion mining or emotion AI) refers to the use of natural language processing, text analysis, computational linguistics, and/or biometrics to systematically identify, extract, quantify, and study affective states and subjective information. *Transfer learning* is a Machine Learning research problem that focusing on storing knowledge gained while solving one problem and applying it to a different but related problem.

In this project, we want you to create a neural sentiment classifier completely from scratch. You first *train* it on one type of dataset and then *apply* it to another related but different dataset. You are expected to make use of concepts that you have learnt in the lecture. The project is divided into three tasks, the details of which you can find in the next pages.

Repository

We have created a Github repository for this project. You will need to -

- *fork* the repository into your own Github account.
- update your *forked* repository with your code and solutions.
- submit the report and the link to your public repository for the available code.

Distribution of Points

The points in this project are equally distributed among the three tasks. You will able to score a maximum of 10 points per task, resulting to a total of 30 points in the entire project. How the 10 points are allotted for each task, is mentioned in the guidelines.

Task 1: Word Embeddings

Neural networks operate on numerical data and not on string or characters. In order to train a neural network or even any machine learning model on text data, we first need to convert the text data in some form of numerical representation before feeding it to the model. There are obviously multiple ways to do this, some of which you have come across during the course of this lecture, like the one-hot encoding method. However, traditional methods like one-hot encoding were eventually replaced by neural Word Embeddings like Word2Vec [1, 2] and GloVe [3]. A word embedding or word vector is a vector representation of an input word that captures the meaning of that word in a semantic vector space. You can find a video lecture from Stanford about Word2Vec here for better understanding.

For this task, you are expected to create your own word embeddings from scratch. You are supposed to use the HASOC Hindi [4] sentiment dataset and train a neural network to extract word embeddings for the data. The unfinished code for this task is already in place in the corresponding Jupyter notebook which you can find in the repository.

To Do

- Follow the instructions in the notebook, complete the code, and run it
- Save the trained word embeddings
- Update your repository with the completed notebook

Task 2: Sentiment Classifier & Transfer Learning

In this task you are expected to reproduce *Subtask A* from the HASOC paper [4] using the Hindi word embeddings from Task 1. Then, you will apply your knowledge of transfer learning by using your model from Task 1 to train Bengali word embeddings and then use the trained classifier to predict hate speech on the Bengali dataset. The data is already included in the repository.

You are expected to read some related research work (for example, encoder-decoder architecture, attention mechanism, etc.) in neural sentiment analysis and then create an end-to-end neural network architecture for the task. After training, you should report the accuracy score of the model on test data.

a) **Binary neural sentiment classifier**: Implement a binary neural sentiment classifier for the Hindi section of the corpus. Use your word embeddings from Task 1 for that. Report the accuracy score.

b) **Preprocess the Bengali data**: Split off a part of the Bengali corpus such that it roughly equals the Hindi corpus in size and distribution of classes (hatespeech/non-hatespeech). Then, apply the preprocessing pipeline from Task 1 to the new data. You can deviate from the pipeline, but should justify your decision.

c) **Bengali word embeddings**: Use the model you created in Task 1 to create Bengali word embeddings.

d) **Apply** classifier to Bengali data, and report accuracy. Retrain your model with the Bengali data. Report the new accuracy and justify your findings.

Task 3: Challenge Task

In this third and final task of this project, you are expected to -

- Read multiple resources about the state-of-the-art work related to sentiment classification and analysis
- Try to come up with methodologies that would possibly improve your existing results
- Improve on the 3 accuracy scores from Task 2.

Note: The task here should be a change in the model architecture, data representation, different approach, or some other similar considerable change in your process pipeline. Please note that although you should consider fine-tuning the model hyperparameters manually, just doing that does not count as a change here.

General Guidelines

- For task 2, 10 points are distributed among implementation, results, analysis, and report as 5, 2, 2, and 1 points. For task 3, the distribution is 3, 1, 3, and 3 points.
- You are not allowed to use ready-made libraries like Hugging Face.
- You are allowed to use convenience methods for data loading & preprocessing from packages like scipy, numpy, pandas, sklearn, NLTK. If you use other packages, provide a reference and justify it.
- Plagiarism will be penalized and can eventually lead to disqualification from the project and the course. Most importantly we will check for plagiarism within groups. If we see any clear indication of plagiarism among groups both of the groups will take a 0 for the whole project. Discussion with groups is allowed but in terms of concepts but not directly with code
- Cite any resources that you found to be helpful.
- You are expected to provide a separate notebook for each task. However, the notebook should only contain runtime code. Functions or classes you write should be in separate python scripts (.py) that are imported into the notebook of that task.
- Your code should be sufficiently commented so that we can grade it easily. Not providing documentation can lead to your code not being graded.
- Write a well documented academic report. The report needs to be **4-8** pages (without long with a NIPS format. You can have a look at Latex versions or in other formats. We expect from you a .pdf file from you. The way how you divide it is up to you but we roughly expect to have introduction,methodology,results and conclusion sections. Of course you will have to cite every source that you use.
- The main focus of our grading will be your observations and analysis of the results. Even though you might obtain bad results make comments on what could have gone wrong.

Contact

In case you encounter any problems or have any doubts regarding this project, please feel free to contact us (Julius Steuer, Sourav Dutta) on *MS Teams*.

Submission instructions

The following instructions are mandatory. If you are not following them, tutors can decide to not correct your exercise.

- You are required to submit the final project as a team of two students.
- You should submit a detailed report and implementation in a zip file. Link to the archived repository should also suffice.
- Make sure to write the MS Teams username, matriculation number, and the name of each member of your team on your submission.
- If you have any trouble with the submission, contact the tutors **before** the deadline.

References

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26:3111–3119, 2013.

[2] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[3] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[4] Thomas Mandl, Sandip Modha, Prasenjit Majumder, Daksh Patel, Mohana Dave, Chintak Mandlia, and Aditya Patel. Overview of the hasoc track at fire 2019: Hate speech and offensive content identification in indo-european languages. In *Proceedings of the 11th Forum for Information Retrieval Evaluation*, pages 14–17, 2019.