**CS 590 NLP**

**EC2**

**MLP + Word2Vec**

**Due 04/06 11:59 pm**

**Overall Goal:**

In this extra credit you will train your own word2vec embeddings and combine them with simple neural networks for classification.

|  |
| --- |
| **Useful python packages:** |
| For word2vec, you may find the gensim package useful: <https://radimrehurek.com/gensim/models/word2vec.html>  Note that gensim has built in pretrained word2vec embeddings so you should choose 1 or experiment with multiple in this homework. (<https://radimrehurek.com/gensim/models/word2vec.html#pretrained-models)>  Gensim also has nice built-in functions for working with word2vec data. |
| For the Multi-Layer Perceptron, there is a sklearn package:  <https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html>  You may also implement this manually, but this would require also implementing the loss algorithm and stochastic gradient descent for training, so I would recommend against this. |

|  |
| --- |
| **Dataset** |
| The dataset that will be used is the toxic comment dataset (same as HW3-4). This consists of toxic comments, so be cautioned when viewing the dataset.  You can find the training and test sets here: <https://www.kaggle.com/competitions/jigsaw-toxic-comment-classification-challenge/data>  Note for the word2vec tasks you will only need to worry about the toxic label (toxic=1 or 0).  The training set (**train.csv)** will be used for training the LM, while the test set (**test.csv**) and labels **(test\_labels.csv)** will be used for testing and analyzing your models.  Take time to understand how the training set and test set are laid out. |

|  |
| --- |
| **Final Report** |
| Just like in the previous homework, you will be documenting observations and results in a final report. This is the best way to show your thought process throughout the homework, so it is best to update it as you go along and then refine at the end, rather than try and write all at the end. |

|  |
| --- |
| **EXTRA CREDIT (20 pts): Improving Pretrained word2vec embeddings** |
| You will fine-tune one of the built in word2vec embeddings on the training text of the toxic data. Essentially, you will be aiming to make the word2vec embeddings better representatives of the toxic data. You will then be using these updated embeddings in your MLP model to see if there are any improvements.  **NOTE:** **Extra credit will not be counted for any late submissions!**  You will create 1 required functions for this task (you may create more functions for your own use, but you need to at least create these two as specified):  Note, you will need HW6 implemented to compare so make sure you first complete that!  **update\_embeddings(path\_to\_train\_file)**  This method fine-tunes a pretrained word2vec embedding (your choice) on the training text and then saves the fine-tuned embeddings. The function should then return the path to the fine-tuned embeddings.  **The format for the train file should follow the same format as the training data file!**  **You should also modify your MLP function to take in an argument for the word2vec embeddings to use.**  **You should hand in a separate file for the EC.**  **Don’t mix HW6 and EC’s code! (You may copy HW6 code to your EC submission).**  Once these functions are implemented and modified use them to accomplish the following tasks.   1. Train one set of pretrained word2vec embeddings on the OLID training text. 2. Test one of your trained MLP models on the test set producing scores for both the original pretrained embeddings and the fine-tuned embeddings. (Note that you will need to train one MLP on the fine-tuned and one on the pretrained.) 3. Compare the overall accuracies of the pretrained embeddings MLP to the fine-tuned embeddings MLP. Are the accuracies better, worse, or the same? Is this surprising or not? What might be causing the results? 4. Write the analysis and comparisons to your report file. This report is how you can demonstrate that you are thinking through the problem and give you practice for your final project. 5. You should provide a link to your fine-tuned embeddings on your final report. Since the embeddings might be too large to hand in on brightspace, you should find some place to host them (onedrive, google drive, etc.)   **Note, there should be some apparent differences between the embeddings themselves. A simple way to check is by looking at specific words and the most similar words to them in both spaces. If you can’t find differences, then there is most likely a problem with your fine-tuning process.** |

|  |
| --- |
| **Additional Tips/Guidance** |
| 1. The accuracy you achieve will not determine your final grade for this homework. Rather your thoughtfulness in approaching and analyzing your models. This means that lack in analysis will receive lower scores as indicated in the grading scale. **However, not addressing errors in implementation will cause loss of points. (e.g. if you get the same results for different embeddings, you probably are making a mistake!)** 2. Note that preprocessing the text is up to you. This assignment is not specifically evaluating preprocessing like the first assignment, but here you have a chance to practice any preprocessing for your final project and future assignments. (**You should be doing some preprocessing, or else you’ll be purposely setting your ML model up to fail.**) 3. Get familiar with the gensim and sklearn libraries on your own on small sets of data. Just copying and pasting the code without fully understanding it will end up being detrimental as you may not be able to accurately give proper analysis and observations of results (especially for the MLP model where you are expected to make improvements/changes).    1. Do not just copy and past the sklearn class code. You may use portions from the sklearn classes, but too much unused code from classes will result in lost points due to poorly written code. Experiment with the code and only import what you need. 4. DO NOT SHARE CODE. I have linked sklearn libraries which will be useful for this assignment. You may also come to me to discuss/figure things out. 5. Start the assignment early. You will have 2 weeks to complete this assignment, which is plenty of time if you start early any familiarize yourself with the libraries. **IF YOU WAIT UNTIL THE LAST MINUTE TO START, YOU WILL BE LESS LIKELY TO DO WELL.** I won’t be granting any additional extensions so each late assignment with follow the late grade policy. 6. Ask questions/approach like a researcher. Think like this is your chance to explore NLP models and analyze their effectiveness. |

**Grading**

Assignment will be graded as follows:

|  |  |
| --- | --- |
| **Description** | **Points** |
| W2V/MLP tasks and implementation | 10 |
| Report | 10 |
| **Total:** | **20** |