CS 4400 - Solution Outline

Cameron Potter

April 2021

1 Code Link

https://github.com/CameronGPotter/CS4400_FinalProject_Repo

2 Solution Outline

The solution includes five steps: (1) Data reading (2) Preprocessing Data, (3) Defining the Model, (4) Model training, and (5) Evaluating the Model and Generating output. We choose an existing architecture known as deepmatcher and based our model off their paper provided here:
http://pages.cs.wisc.edu/~anhai/papers1/deepmatcher-sigmod18.pdf

2.1 Data Reading

This step simply involved reading the .csv files to use them as dataframes in the code.

2.2 Preprocessing Data

Here, we rename the columns of the dataframes to work with our chosen architecture. Then, we add an id column to the labeled data. Next, we merge the labeled data with our left and right tables to get the attributes. Finally, we split the labeled data into our train, and validation data with a 0.7499, 0.25 split. The extra 0.001 goes into a test set, but since the test set will be the F1 score on the ground truth, we leave it out to get more data to train with. We then run the process data method within the deepmatcher library. We ignore the id columns here so the model does not train by matching left and right id numbers.

2.3 Defining the Model

After lots of testing, we decide to use the hybrid model defined in the deepmatcher library. It returns the highest F1 score out of the avaliable models.

2.4 Model Training

We attempt to tune the hyperparamters of the model to maximize the F1 Score. All are left as default excluding pos_neg_ratio which we set to 8.65 and epochs which is set at 15 due to overfitting of the dataset.

2.5 Evaluating the Model and Generating Output

Here, we run the evaluation of our model and use it to generate our output.csv file. We return all pairs with match_score greater than 0.5. We block by the title on the left table and brand on the right table to help decrease the number of pairs needed to test. These pairs are sent to the output.csv file.