**REPORT - DIRECTED RESEARCH – CSCI 590 (FALL 2015)**

**SPEECH CONTENT ANALYSIS OF SPEAKERS IN DEBATES, APPLIED TO THE MACHINE LEARNING PROBLEMS OF DEBATE WINNER PREDICTION AND PREDICTING EARLY ONSET OF SCHIZOPHRENIA IN HIGH RISK YOUTH**

NAME: ABHISHEK SHARMA RAMACHANDRA

USC ID: 7960-7575-87

MAIL ID: [rama055@usc.edu](mailto:rama055@usc.edu)

ABSTRACT:

The goal of the work done during the course of the project was to analyze the content of statements made by speakers in a debate and to apply the analysis done to solving the machine learning problems of predicting the winner in debates and predicting early onset of schizophrenia. The data set used was the compilation of the set of transcripts for opening and closing statements of speakers for 36 debates along with the user comments posted in the “(<http://intelligencesquaredus.org/)>” webpage for these debates. These debates are conducted by the Intelligence Squared organization. A word corpus was built using all the words in the transcripts as well as the comments posted by users for each debate. A two layer neural-net called Word2Vec was used to transform the word corpus into feature vectors of common size for the words.

Word2vec is a two-layer neural net that processes text. Its input is a text corpus and its output is a set of vectors: feature vectors for words in that corpus. While Word2vec is not a deep neural network, it turns text into a numerical form that deep nets can understand. The purpose and usefulness of Word2vec is to group the vectors of similar words together in vectorspace. That is, it detects similarities mathematically. Word2vec creates vectors that are distributed numerical representations of word features, features such as the context of individual words. It does so without human intervention. Given enough data, usage and contexts, Word2vec can make highly accurate guesses about a word’s meaning based on past appearances. Those guesses can be used to establish a word’s association with other words (e.g. “man” is to “boy” what “woman” is to “girl”). The output of the Word2vec neural net is a vocabulary in which each item has a vector attached to it, which can be fed into a deep-learning net or simply queried to detect relationships between words. These vectors serve as features for the compiled data set in this project.

Additional feature vectors such as first order and second order cosine distance between the mean of the vectors corresponding to words that belong to the same line are introduced. The first order cosine distance is the distance between a line ‘x’ that belongs to a speaker’s opening or closing statement for a given debate and the line x+1, similarly, the second order cosine distance is the distance between line ‘x’ and line ‘x+2’. The cosine distance between two unrelated lines is 1. An SVM classifier is trained to predict debate winners using the abovementioned feature vectors and a prepared csv file which contains both the data set and the test set.

SCRIPTS:

1. extract\_comments\_from\_url.py: This script extracts the user comments from the webpage for a given IntelligenceSquared debate url and puts it into a file called comments.txt.
2. split\_text\_into\_lines: This script goes through all transcripts of opening and closing statements for each debate, splits the file into paragraphs that are separated by “EOP” and splits the paragraphs into sets of lines that are seperated by an empty line. For a file ‘x’, this script creates a file ‘split\_x’ which has the split paragraphs for the given file.

1. get\_all\_lines.py: This script iterates through all ‘split\_x’ files and comment file for every debate and stores the set of lines present in each of them in a file called ‘all\_lines’. 'all\_lines’ is a list of lines where each line is a list of words present in it. This is the expected input format for word2vec.py.
2. build\_word\_corpus: This script stores identification information for each word that is present in either the comment files or the split transcript files. For each word, it stores debate ID, speaker ID, opening or closing statement, paragraph number, line number and it’s position within the line . For a word that is only present in a comment file, it stores only debate ID. This information is stored as a dictionary in a file called “word\_corpus”.
3. Word2vec.py: This script takes the words stored in “all\_lines” and feeds it as input to the Word2Vec text classifier. This trains a model based on the parameters specified in the script (Models were trained for vector size 10,20,50,100 and 300 using this script). The model is stored in “all\_words.model” .
4. Compute\_mean\_vector.py: This script uses the feature vectors produced by the word2vec.py script and computes the mean vector for all words belonging to every line encountered so far. (split transcript + comments). It also generates a csv file which has the mean vectors for each line along with the debate ID, speaker ID, closing or opening statement and line number information. This information is stored in a file called ‘avg\_vec.csv’.
5. TrainSVM.m: This script trains an SVM classifier using the data obtained from the csv file. The feature vectors used are the mean vectors (already present in csv), first order cosine distance and second order cosine distance (computed in this script for every line by the same speaker in the same statement for the same debate). It also tests for significant differences between these features for winners and losers. The trained SVM classifier attempts to predict debate winner for the test set.