# CMPT-741 Course Project – Sentiment Analysis base on CNN

## Task Description – Version 2 (October 10 2016)

A note: this first version contains some basic information of the project description. A refined version will be posted as the project progresses and an email alert will be sent if any.

## [Introduction]

In this project, you are asked to perform ***Sentiment Analysis***, a broadly-studied research problem in Natural Language Processing (NLP) area. Briefly speaking, ***Sentiment Analysis*** is to build a classifier that predicts the sentiment of natural language sentences (i.e., positive/negative labels). Here's an example of training data with two sentences. From a given training data, a good classifier will capture important features that determine the label of sentences and use such features to predict the labels of new sentences with unknown labels. You are highly recommended to start this project as early as possible. Counting on the last 1 or 2 week to finish the project will be unrealistic.

| Data | Label |
|---|---|
| This is a film well worth seeing, talking and singing heads and all . | positive or 1 |
| Nothing scary here except for some awful acting and lame special effects. | negative or 0 |

## [Requirements]

In order to build the classifier, you are asked to use ***Convolution Neural Networks (CNN)*** instead of other data mining models. [4] and [2], a blog discussion of [4], contain detailed materials on using CNN for sentiment analysis. You must use ***Matlab*** for coding, and for the ***CNN*** part, must use the library *MatConvNet* (i.e., functions vl_nnconv(), vl_nnpool(), vl_nnrelu(), vl_nnconcat(), vl_nnloss()), with which you can build up your CNN models and learn the neural network parameters easily and efficiently.

This is a group project with each group having two students working together. We do encourage you to form a group, instead of working alone.

## [Rules]

**Dataset:** We will use a ***Movie Review Dataset*** in this course project. You will be given a set of 6000 labeled sentences contained in the file *train.txt*. Each sentence has either label 1 (positive) or label 0 (negative). An example of a sentence in the *train.txt* file is:

*1::the satire is unfocused , while the story goes nowhere::0*

Here, 1 is the sentence ID and 0 is the label. The content of the sentence is everything within the pair of ::. The tokens, i.e., words and punctuation marks, are separated by space. Note that you do not need

to do further preprocessing of the data set (such as removing stop words, etc). You should use the given sequence of tokens (separated by space) as the input example to your model.

You should reserve some portion of this data as the testing set to evaluate the classifier learnt from the remaining examples. The goal is to learn a good classifier that can predict the label of future sentences without unknown labels.

For your reference, we provide a function read_data() to read the file train.txt stored in the same folder. We also provide a template structure of code in the file train_model.m

**Evaluation and Grading:** The evaluation is based on another 1000 sentences with known labels, i.e., called *evaluation_set.txt*, to compute the ***accuracy*** of your model. This set will not be released until you are evaluated. The total 100 points of this project consist of:

(1) 40% - A 4-page report that describes key steps of your approach, discussion on techniques used and choices made, and the code. The report is due on Dec 7 2016.

(2) 60% - Prediction accuracy: There will be a demo session for evaluating the accuracy of your model. The demo time is Dec 6 (Tuesday),

3pm-7pm. A doodle poll will be set up for you to pick your preferred time slot. Each slot is about 5-7 mins long. You will bring your own laptop, compile your file, read an input file given to you on-the-fly, and write the result to the output file.

The input file for evaluation, called *evaluation_set.txt,* contains 1000 random sampled sentences with labels removed. Your model will read each sentence and predicts its label and output the results into an output file. Then our program will read this output file to evaluate the accuracy of your result. The formats of the input/output files are specified below and you must follow strictly these formats; otherwise, the files cannot be read by either your program or our program.

Each correct prediction will earn 0.06 points. So the perfect sore for predicting 1000 sentences is 60 points.

## [Input/Output format]

Input file format

Each sentence is a line. An example is

*1::the satire is unfocused , while the story goes nowhere::*

Here 1 is the ID of the sentence. Note that the true label of the sentence is removed. The pair of *::* encloses the actual content of a sentence. Tokens in the sentence are separated by space.

Output file format

1 0

2 1

3 0

4 1

On each line, the first number is the sentence ID and the second number is the predicted label (either 1 or 0).

## [Resources]

The following links contain information on how to use ***CNN*** to perform sentiment analysis, how to install and use ***MatConvNet***. These resources should help you to do these quickly.

1. Matlab: matlab cookbook
2. CNN for NLP: understanding convolutional neural networks for nlp
3. Implement CNN on MatConvNet: CNN practical
4. The original paper: Convolutional Neural Networks for Sentence Classification, EMNLP 2014

For project related questions, please contact the TA, "Jiaxi Tang" jiaxit@sfu.ca. His office hours: Thursday 2-3pm at ASB 9808.