ASSIGNMENT 1: SENTIMENT MINING and DOMAIN ADAPTATION

Motivation: The motivation of this assignment is to get practice with text categorization using Machine Learning algorithms. This assignment will proceed in four parts. Each part is worth equal points.

PARTS I-III

Problem Statement: The goal of the assignment is to build a sentiment categorization system for product reviews from annotated data. The input of the code will be a set of annotated product reviews from the domain of audio records.

Training Data: We are sharing a training dataset of product reviews. Each data point has review text (including the title), and a rating. Ratings are between 1 and 5. For the purpose of this assignment, we will consider 1-2 to be negative, 4-5 to be positive, and 3 to be neutral. This is for evaluation, though, you are welcome to use the fine-grained labels as appropriate.

The Task: You need to write three classifiers that predict for a given new review (from the same domain) its sentiment polarity. In Part I, you must develop a non-neural classifier. Ideas include Naïve Bayes, logistic regression, nearest neighbor, rule/lexicon-based systems and so on. In Part II you will build a neural classifier using CNN, LSTM, Paragraph vectors, and any other such method. In Part III you will build your best classifier – this might be a combination of techniques from Part I and II or might simply be the best classifier from the first two parts.

For Part I, as a baseline algorithm, you could use all words as features and learn a classifier (naïve Bayes or logistic regression). To improve your system performance over the baseline I list a few ideas below.

1. Try changing the classifiers. Try SVMs, random forests or even multilayer perceptrons.
2. If you use logistic regression try playing with regularizers – try L1 instead of L2. You could also implement a different feature selection procedure.
3. Try to work with the features. You could lemmatize. You could get rid of stop words and highly infrequent words. You could use tfidf-based weighting.
4. Try to use existing sentiment resources discussed in class. Examples: SentiWordnet or General Inquirer.
5. You could work with bigrams (in addition to unigrams).
6. You could define new features, like you could pos-tag each word and use the tagged word as a feature instead of the original word. You can use presence of capitalization or all caps as features.
7. Your idea here…

For Part II, as a baseline algorithm, we suggest training a simple ParagraphVec model and using a 1 or 2 layer network for sentiment classification. You are not allowed to use other features (or other resources) in this model. You are allowed to use product ID if you wish, since that is part of training data. For extensions

8. You could try to learn a CNN
9. You could try to learn an LSTM/GRU based model
10. You could try a combination model

For Part III, you must first choose the best architecture. It could be an SVM/Logistic model with neural features, or a neural model with sentiment resources (and other features) OR a hybrid model that combines predictions made by different algorithms (neural and non-neural).

Methodology and Experiments:

You are being provided two sets – audio_train.json and audio_dev.json. You must do hyperparameter tuning on the dev. Once you find the best hyperparameters, we recommend training on the whole training+dev dataset.

As you work on improving your baseline system document its performance. Perform error analysis on a subset of data and think about what additional knowledge could help the classifier the most. That will guide you in picking the next feature (or model component) to add.

Test Format:

Your final program will take input a set of reviews in the same format as training, without the rating. Your program will output predictions (1 3 or 5) one per line – matching one prediction per review.

PART IV

Problem Statement: The goal of the assignment is to build a product sentiment mining system for a different domain, based on the training data given to you for audio recordings (out-of-domain), plus a small (about 1000) annotated data points of the new domain. The goal is to see how effectively you can transfer to a new domain.

Adaptation Data: We are sharing ~1000 annotated reviews from a new domain to test your domain adaptation system. At eval time, we will share ~1000 annotated reviews from a different domain. And you will be expected to submit a sentiment mining system in two days (48 hours).

The Task: You can always use your system from Part III as the baseline system. To improve your system performance:

1. Simply combine both datasets and relearn.
2. Upweight the in-domain training data in #1 above.
3. Use ideas from bootstrapping as discussed in class to induce domain-specific features.
4. Use PMI of top words in out-of-domain training data with those in in-domain-training data to upweight the features in in-domain data. Similar papers have recently come out of Pushpak Bhattacharya's lab (IIT Bombay). You may try those or other ones.
5. Use fine tuning of parameters for neural transfer
6. Use latest methods for neural transfer. You will need to dig and read references.
7. Your idea here… (feel free to search for ideas on the Web).

What to submit?

1. Different deadlines will be announced as the course evolves. The first deadline will be on 9th March 11:55 PM for the submission of Part I system.
   Submit your code in a .zip file named in the format **<EntryNo>.zip.** Make sure that when we run "unzip yourfile.zip" a new directory is created with your entry number (in all caps). In that directory, the following files should be present.
   compile.sh
   run.sh
   writeup.txt

   You will be penalized if your submission does not conform to this requirement.

   Your code will be run as ./run.sh inputfile.json outputfile.txt. The outputfile.txt should only contain a sequence of numbers (1 3 or 5), one per line, with total number of lines matching the number of reviews in inputfile.txt. If your code doesn't conform to the requirements you will lose 20% of the credit.

   Your code should work on our Baadal servers with 2GB memory. You will be penalized for any submissions that do not conform to this requirement.

2. The writeup.txt should have first line that mentions names of all students you discussed/collaborated with (see guidelines on collaboration vs. cheating on the course home page). If you never discussed the assignment with anyone else say None.

After this first line you are welcome to write something about your code, though this is not necessary.

Evaluation Criteria

(1) Each part is worth 50 points.
(2) Bonus given to outstanding performers.

What is allowed? What is not?

1. The assignment is to be done individually.
2. **You may use only Python (and PyTorch) for this assignment.**
3. You must not discuss this assignment with anyone outside the class. **Make sure you mention the names in your write-up in case you discuss with anyone from within the class outside your team.** Please read academic integrity guidelines on the course home page and follow them carefully.
4. Feel free to search the Web for papers or other websites describing how to build a sentiment classifier. However, you should not use (or read) other people's sentiment mining code.
5. You can use any pre-existing ML softwares for your code. Popular examples include Weka (http://www.weka.net.nz/), Python Scikit (http://scikit-learn.org/stable/). However, if you include the other code, use a different directory and don't mix your code with pre-existing code.
6. We will run plagiarism detection software. Any team found guilty will be awarded a suitable penalty as per IIT rules.
7. Your code will be automatically evaluated. You get significant penalty if it is does not conform to output guidelines. Make sure it satisfies the format checker before you submit.