SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

50.040 Natural Language Processing, Summer 2019
Homework 1

Due 7 June 2019, 5pm

This homework will be graded by Richard Sun

Word embeddings are dense vectors that represent words, and capable of capturing semantic and syntactic similarity, relation with other words, etc.

Word2Vec is one of the most popular techniques to learn word embeddings using shallow neural networks. It was developed by Tomas Mikolov in 2013 at Google.

Generally, there are two methods to evaluate the quality of word embeddings. One is intrinsic evaluation, and the other is extrinsic evaluation. In intrinsic evaluation, the similarities between words are explored whereas in extrinsic evaluation, downstream tasks are executed based on word embeddings.

In order to finish the following tasks, you need to download the text8 dataset and put it under the "data" folder. The text8 dataset consists of one single line of long text. Please do not change the data unless you are requested to do so.

Task 1: Preprocess the dataset (6 points) Find out the context and target pairs in a given sentence "I am interested in NLP.", assume the window size is 1. We need to preprocess the dataset, tokenize the text into words, create word batches for subsequent training.

Task 2: Train our own embeddings (2 points) We will train our own word embedding on text8 dataset, which contains a single line of text. We will use Python gensim package to facilitate the procedures. Set the word embedding size as 100, select Skip-gram model.

Task 3: Visualize word embeddings (8 points) Visualization is often employed in analyzing word embedding. Before visualization, we need to project the 100-dimensional word embeddings to 2-dimensional points.

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components. PCA can be used to reduce dimension by selecting several principal components.

Our tasks:

- Select 500 most common words from the vocabulary of text8 dataset
- Implement a PCA algorithm using numpy package and built-in functions, do not use other third-party tools.

- Project the embeddings of the 500 words to 2-dimensional points, plot those points.
- Try to find patterns in your visualization.

Task 4: Intrinsic evaluation (4 points) In this exercise, we'll do some interesting explorations on our trained embeddings using gensim. Our tasks:

- Find out five most similar words for each of "dog", "cat", "eat".
- A popular choice for intrinsic evaluation of word vectors is to explore the performance in completing word vector analogies, assume there are two word pairs, a:b, c:d, ideally, their embeddings satisfy a rule $x_a - x_b = x_c - x_d$. For instance, queen – king = actress – actor. Now find out which word will it be in woman - king = man - ?.

Task 5: Extrinsic evaluation (10 points) Apart from intrinsic evaluation, the quality of word embeddings can also be evaluated by downstream tasks such as sentiment analysis, which aims to detect the sentiment polarity of a sentence. For instance, the sentence "I like the movie very much" is positive, whereas "I was very disappointed with my new phone" is negative.

We will build a sentiment analysis pipeline and train a classifier based on GRU using embeddings we trained above, the inputs will be sentences in the dataset, the outputs will be the sentiment polarities.

Note, for this exercise, we need to use pytorch package, if you are not familiar with pytorch, you can refer to tutorials. The sentiment dataset is under the folder "data".

Our tasks:

- Tokenize sentences into words, map each sentence to a sequence of embeddings(we trained above).
- As lengths of sentences are different, we need to pad the sentences in a batch to the same length and stack their sequences of embeddings into a tensor.
- Complete a GRU classifier and train the classifier using pytorch on training dataset.
- Evaluate the performances of the classifier on testing dataset. The accuracy should be above 0.8.

Requirements:

- Complete answers and code in the "homework 1.ipynb" file. Submit the file before the due on eDimension system.
- Write down names and IDs of students with whom you have discussed (if any).
- Follow the honor code strictly.