PRACTICAL 1

Due on March 16, 2018 (23:59:59)

Instructions. Theddd main goal of this practical is to make you understand the basics of neural network design and training, and you will also get familiar with Google's TensorFlow deep learning framework. In this practical, you will use the `word2vec` [1] implementation included in the Tensorflow code repository and train a multi-layer perceptron to learn distributed representations of words on TED and Wikipedia datasets. After training, you will analyze and visualize the learned embeddings t-SNE [2], a popular dimensionality reduction method, and then utilize the word embeddings for sentiment classification using the DAN approach [3].

Getting set up

The first step is to install TensorFlow on your own machine. To download and install TensorFlow, simply follow the instructions on the Tensorflow website.

Part 1 - Learning distributed word representations[1]

You will use `word2vec` model [1] and train your own word vectors with stochastic gradient descent (SGD). To make your life easier, you can use the basic implementation (`word2vec_basic.py`) provided on the Tensorflow website but you have to try different settings and play with the parameter values to explore their influence on the model behavior on the resulting representations. Here, to train your word embeddings, you will employ TED corpus (link provided on the course website).

In your experiments, perform the following experiments and explain your findings.

- Play with the batch size,
- Play with the embedding size, i.e. the number of dimensions in the distributed representation
- Play with skip window size which determines how many words to consider to the left and to the right.

Part 2 - Qualitative analysis

Once you trained a `word2vec` model, you can use the learned embedding space to analyse how similar or intuitively related two given words are. In this part of the practical, you will

[1] Adapted from the practical developed by Brendan Shillingford, Yannis Assael, Chris Dyer for Oxford Deep Learning for NLP course.

train models on two different corpora, TED and Wikipedia datasets (links to these datasets are provided at the course website). You will then evaluate each one of your trained models by returning a list of the most similar words to the following words:

- "`man`",
- "`computer`",
- "`laugh`",
- "`enjoy`"

Additionally, you will find a few more words with interesting and/or surprising nearest neighbours by your own.

Part 3 - Word clusters

In this part of the assignment, you will analyze the learned embedding spaces from Part 2 via t-SNE [2] which has been shown to be an effective dimensionality reduction method. Through the visualization obtained from t-SNE, you will try to determine certain word clusters which contains intuitively similar word groups.

Part 4 - Sentiment Analysis

Lastly, you will implement a simple sentiment classifier using the Deep Averaging Network (DAN) proposed in [3]. The model is based on the following three steps:

1. Take the vector average of the embeddings associated with the words in the inputs
2. Pass that average vector through one or more feed-forward layers
3. Perform linear classification on the final layers representation

Here, you will use Stanford Sentiment Treebank dataset but note that in this dataset, the sentiment levels are originally represented with real values. Hence, you need to discretize these values into the following five classes:

- 0: "very negative" (≤ 0.2),
- 1: "negative" (≤ 0.4),
- 2: "neutral" (≤ 0.6),
- 3: "positive" (≤ 0.8),
- 4: "very positive" (> 0.8)

In your experiments, perform the following experiments and explain your findings.

- Play with the number of layers,
- Try with embeddings trained on different corpuses

What to turn in

For Part 1, provide computation graph of your models, perform the aforementioned experiments and report your results by plotting loss curves. For each experiment, explain your findings. Put into your report any other information that supports your findings and explorations.

For Part 2, provide your results and explain findings by comparing the retrieval results of each model with each other. For each experiment, discuss the overall effect of each choice of parameters.

For Part 3, find a an interesting cluster in the t-SNE plot. For this task, it is recommended to use Tensorflow's Embedding Projector tool.

For Part 4, provide the classification accuracy values on the training, validation and test splits. Report your findings by plotting loss curves for each configuration.

Your reports should be prepared using LaTeX!. We will be using NIPS style for that. Here is a link to the zip file containing the corresponding template and style files. You can submit your reports by e-mail.

Grading

The practical will be graded out of 100 points: 0 (no submission), 20 (an attempt at a solution), 40 (a partially correct solution), 60 (a mostly correct solution), 80 (a correct solution), 100 (a particularly creative or insightful solution).

Late Policy

You may use up to five *slip days* (in total) over the course of the semester for the three practicals you will take. Any additional unapproved late submission will be weighted by 0.5 and no submission after five days will be accepted.

Academic Integrity

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given practical, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudocode) will not be tolerated. In short, turning in someone else's work, in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.

References

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[2] Laurens van der Maaten and Geoffrey E. Hinton, "Visualizing high-dimensional data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[3] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daume III, "Deep unordered composition rivals syntactic methods for text classification," in *ACL*, 2015.