Neural Computation (VS 265), Problem Set 5

Due date: November 8, 3:30pm

Fall 2022

General guidelines:

- We are grading problem sets anonymously. **Include your student ID in the submission, but do not include your name.**
- You may work in small groups of 2-3. Note that you are responsible for writing up and submitting your submission individually.
- You are expected to attach any code you used for this assignment but will be evaluated primarily on the writeup.
- If you are including animations as part of your submission, please attach these files separately in your submission (it should not be necessary to run any code to view your animations).

Part 1: Multi-layer Perceptron

In PS1, we gave an example of learning a linear decision boundary for a simple problem. We turn to a case with a non-linear decision boundary given by this dataset.

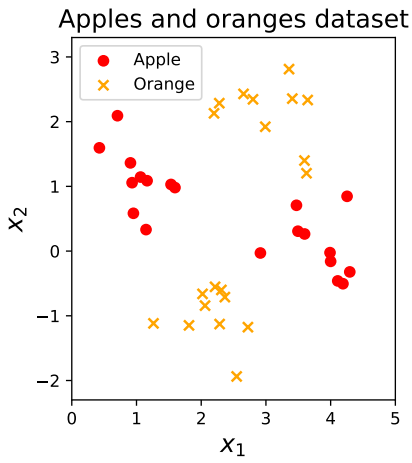

Figure 1: Dataset for Part 1.

i. Train a multi-layer neural network (using the backpropagation algorithm) to discriminate perfectly between the two classes. Your neural network should have one hidden layer. Report the accuracy of your classifier as a function of epochs (as done in Problem Set 1).

ii. Plot the decision boundaries learned by your network:

- *In the input space:* behind the scatter plot of the dataset, color code which regions of the input space are encoded as oranges and which are encoded as apples.
- *In the hidden unit space:* plot each data point in terms of its hidden unit activations. Then color code the decision boundary in the hidden unit space.

Part 2: Semantic Vectors

In this assignment, we will work with data-driven models for forming representations of words (commonly referred to as embeddings). Given a corpus of text, the goal is to form a high-dimensional vector representing a word in the vocabulary. The challenge is to provide vector representations of words where the relationship between vectors mirrors the linguistic relationship between words.

Setup steps (no need to document these in your writeup):

- For the training data, you will use a small text corpus, which is a preprocessed subset of the Gutenberg Dataset.[1] Please download the file containing the text corpus here.
- To evaluate the quality of the obtained embeddings, we will use the Test of English as a Foreign Language (TOEFL) synonymy assessment. Please download the file containing the TOEFL synonymy assessment here.
  - The dataset contains 80 synonym tasks. Each task includes a query word and 4 alternatives. One of the alternatives is a synonym of the query word. If the chosen word is the correct answer (i.e., the synonym), then the score on the overall dataset increases by 1. Thus, the overall score on the TOEFL synonymy assessment is an integer number between 0 and 80. Note that the expected score by chance is 20 (i.e., 25 percent).
- You are provided with a Jupyter notebook (link provided on the course website) that provides code to support loading data, initializing the embeddings, evaluating the obtained embeddings on the synonym tasks, etc.
  - Note that the notebook has a few sections which you will need to implement in the first part of this assignment.
- You have two choices of algorithm to implement, Random Indexing (RI) and BEAGLE.[2] You will need to read the relevant papers provided below to follow the steps on implementing one of these models.
  - RI: Sahlgren 2005, Random Indexing: "An Introduction to Random Indexing" and Sahlgren et al. 2008, "Permutations as a Means to Encode Order in Word Space".
  - BEAGLE: Jones and Mewhort 2007, "Representing Word Meaning and Order Information in a Composite Holographic Lexicon".

Implementation steps (you *do* need to document these in your writeup):

i. Complete the steps marked by a commented *#ToDo* in the notebook. When implementing these steps, you will need to choose RI or BEAGLE as your method for forming the word embeddings. For both approaches, please be sure to implement both the "context" and "order" parts of the embedding.

[1] Usually before building word embeddings, the original text is pre-processed. Common pre-processing steps include disregarding the most frequent words and lemmatization. For the sake of convenience, the provided files contain words that are already lemmatized, so the code does not contain the lemmatization step explicitly.

[2] Technical note: The stem of the code provided in the notebook implicitly assumes that the RI approach will be implemented. So, implementing BEAGLE will be slightly more challenging as you will have modify the provided code around accordingly. For example, instead of the window around the focus word (implemented in the notebook) you would need to consider the whole sentence.

ii. Record and plot the performance of the embeddings on the TOEFL synonymy assessment for several different vector dimensionalities. The dimensionality is up to you, but use different values (e.g., 512, 1024, 2048, etc.). Run simulations several times (e.g., 5) for each dimensionality to obtain average results.

iii. Examine the effect of parameters on model performance (answer the question applicable to your model):

- Random Indexing: how does the size of the window around the focus word affect the results on the TOEFL synonymy assessment?
- BEAGLE: how does the size of n-grams taken into embeddings affect the results on the TOEFL synonymy assessment?

iv. How does accuracy on the TOEFL synonymy assessment change when only context or only order parts of the embedding are used? (Note that on previous parts, both order and context were used.) Provide plots and some interpretation of your results.

v. Look through the TOEFL dataset for examples of where your model performed correctly and incorrectly. Find a specific example of a error on the dataset and suggest one way to improve the word embedding method you implemented to reduce the probability of making this kind of error.