# LING570 Hw11: Word2vec
# Due: 11pm on Dec 14, 2017

A few notes about this assignment:

- The total raw score is 120 points (Q7 is a bonus question). Your final grade for this assignment will be the minimum of 100 and the raw score. In other words, you can get the maximal score, 100 points, even if some of your answers are wrong.
- This is a reading assignment, and the answers to the questions are in the readings, wikipedia pages, and class slides.
- For some questions, I provide a wikipedia page url. But feel free to google the topic and read other related pages.
- The answers to the questions should be pretty short. I leave some space for you to fill out the answers. I also make the latex file available in case you want to add the answers to the latex file directly. In that case, you need to run pdf2latex (or something like that) to generate pdf from the latex file.
- If you prefer to write formulas on paper (instead of typing them with latex or Word), it is ok. You just need to fill out the rest of the assignment, print out the file, insert formulas by hand, scan the paper, and then submit via Canvas.
- Since no programming is required, you only need to submit a single file. Pdf is highly preferred. But if you cannot convert your file to pdf, a jpeg file is ok.
- For Q6-Q7, go over the class slides and read the following:

  **Paper #1:** (Mikolov et al., 2013-ICLR) at https://arxiv.org/pdf/1301.3781.pdf

  **Paper #2:** (Mikolov et al, 2013-NIPS) at https://arxiv.org/pdf/1310.4546.pdf

  **Blog Part 1:** at http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/

  **Blog Part 2:** at http://mccormickml.com/2017/01/11/word2vec-tutorial-part-2-negative-sampling/

  Some details in the two papers may be difficult to follow, but the blogs should help.

- I copied two word2vec packages to dropbox/17-18/570/hw11/:
  - word2vec-dav is better organized, and the code is slightly modified from the original word2vec implementation.
  - word2vec-mccormick is the original word2vec implementation with comments added by McCormick (the author of the blogs).
  - The packages are not required for this assignment. But if you want to dig into the code to see how exactly the models are implemented, the code is not too hard to read.

**Q1 (12 points):** Logistic function: (see https://en.wikipedia.org/wiki/Logistic_function)

**1a (4 pts):** What's a logistic function? Write down the formula.

**1b (4 pts):** What is the standard logistic function? Write down the formula.

**1c (4 pts):** What is the derivative of the standard logistic function f(x)? Write down the formula.

**Q2 (8 points):** Sigmoid function: See https://en.wikipedia.org/wiki/Sigmoid_function

**2a (4 pts):** What is a sigmoid function? Write down the definition.

**2b (4 pts):** What is the relation between sigmoid function and logistic function?

**Q3 (10 points):** Tanh function: (see https://en.wikipedia.org/wiki/Hyperbolic_function and https://en.wikipedia.org/wiki/Sigmoid_function)

**3a (5 pts):** Write down the formula for the tanh function.

**3b (5 pts):** What is the relation between sigmoid function and tanh function?

**Q4 (10 points):** The softmax function:

**4a (5 pts):** What is the softmax function? Write down the formula.
See https://en.wikipedia.org/wiki/Softmax_function

**4b (5 pts):** If a vector x is [1, 2, 3], what is the value of softmax(x)?

**Q5 (18 points):** Matrix: see Sect 1-3 of https://en.wikipedia.org/wiki/Matrix_(mathematics)

**5a (12 points):** Let $A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$ and $B = \begin{bmatrix} 2 \\ 1 \\ 3 \end{bmatrix}$

what is $A \times B$?

what is $B \times A$?

what is the transpose of A?

what is the tranpose of B?

what are the dimensions of B?

what are the dimensions of the transpose of B?

**5b (6 points):** Let $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ and $B = \begin{bmatrix} 2 & 0 \\ 1 & 2 \end{bmatrix}$.

What is $A \times B$?

what is $B \times A$?

Is matrix multiplication communtative?

**Q6 (42 points):** Answer the following questions for the Skip-Gram model. Most of the questions were covered in class. For Q6, assume that the vocabulary has 100K words, and the word embeddings have 50 dimensions.

**6a (5 points):** What is the "fake task" in order to learn word embeddings? That is, for this fake task, what are the input and the output at the **test** time?

**6b (5 points):** How many layers are there in the neural network for solving the fake task?

How many neurons are there in each layer?

**6c (5 points):** Not counting the vector for the input word and the output vector for the output layer, how many matrices are there in the network? What are the dimensions of the matrices?

How many model parameters are there? That is, how many weights need to be estimated during the training?

**6d (5 points):** Why do we need to create the fake task?

**6e (10 points):** For any supervised learning algorithm, the training data is a set of (x, y) pairs: x is the input, y is the output. For the Skip-Gram model discussed in class, what is x? What is y?

Given a set of sentences, how to generate (x, y) pairs?

Notice that my lecture and the blogs give slightly different answers to what y is. You can use either answer. Just specify whether the answer is from my lecture or from the blogs.

**6f (5 points):** What is one-hot representation? Which layer is that used? Why is it called one-hot?

**6g (7 points):** Softmax is used in the output layer. Why do we need to use softmax?

**Q7 (20 points):** Read the two papers and the blogs mentioned at the beginning of the assignment, and answer the following questions:

**7a (3 points):** Based on Section 4 of paper #1, Other than different neural network models, what other factors can affect system accuracy? Name at least three factors.

**7b (5 points):** What is negative sampling? What benefit does it provide?

**7c (5 points):** Why subsamples words? How is that done? Paper #2 and Blog #2 use different formulas. You can choose either one. Just specify which one you use.

**7d (7 points):** How did paper #2 find *phrases*?

Suppose you run 3 passes over the training data to find phrases, how long can a phrase be in theory? That is, what's the maximum length of a phrase that can be found after 3 passes of the training data?

Once the phrases are found, how do you train a model to find words/phrases that are similar to a given phrase?

**Submission:** submit only one file named hw.pdf or hw.jpeg to Canvas.