2. Neural Classifiers

- Gradient descent
    - The word vectors in Word2vec are optimized via gradient descent
    - The general idea is walk a little bit downwards (where "downwards" refers to an area of lower error)
        - The distance walked each iteration is referred to as the learning rate
    - But gradient descent is VERY inefficient
        - Very computationally expensive because the number of words is huge
- Stochastic gradient descent
    - Rather than considering all words each iteration, look at a subset of words
        - This window of words is sampled from the entire corpus
            - Not typically uniformly, higher weight is given to words that occur more frequently in language
    - Repeatedly sample these
    - This process ends up being more performant and more accurate
- Skip-gram negative sampling (SGNS)
    - Working out all the dot products in a softmax function is still expensive
    - SGNS appeared as an alternative
        - Instead of a softmax, fit a logistic regression model
            - If dot product is large, make probability essentially 1
            - If dot product is small, make probability essentially 0
- Co-occurrence matrices:
    - Put the entire list of distinct words in a symmetric matrix (as the corpus comprising both the rows and columns)
    - Algorithm:
        - Select a word
        - Count the frequency of a specific word occurring within X words of this selected word
            - X is typically considered to be the window
        - Represent these counts as a vector (a row / column of the matrix)
    - But co-occurrence matrices require a lot of storage
- GloVe:
    - Goal: unify linear-algebra based algorithms with iterative NN algorithms to mitigate the cons of both
    - Worked well, but this may be due in part to better data