Word Embeddings CBOW vs Skip-gram

By Shuning Zhao
2017/11/09.

Introduction
Word embeddings have been shown to improve the performance of many NLP models by converting words from character arrays to vectors that contain semantic infomation of the word itself. In this task will implement a Continuous Bag of Words (CBOW) version of word2vec - one of the fastest and most commonly used embedding algorithms.

A good introduction to word embeddings can be found in the TensorFlow word2vec tutorial. This section of the code builds on that tutorial.

The aim of this task is to modify the code here so that it uses the continuous bag of words (CBOW) model instead of the skip-gram model. This should produce better embeddings, particularly when less data is available. Furthermore, implementing this change should give a better understanding of both models, and the differences between them.

CBOW vs Skip-gram
Input-Output
The main difference between the skip-gram and CBOW model, is the way training data is presented.

With the skip-gram model, the input is the word in the middle of the context window, and the target is to predict any context word (word that is $skip_window$ words to the left or the right) for the given word.

With CBOW, the input is all the words in the context besides the middle word, and the target is to predict the middle word, that was omitted from the context window.

For example, given the sentence fragment ”the cat sat on the”, the following training examples would be used by skip-gram, with parameters $skip_window = 1$, $num_skips = 2$ - in the form:

{words in context window} (input, target)
{cat sat on} (sat, cat), (sat, on),
{sat on the} (on, sat), (on, the)

While for CBOW the input-output pairs are (note that the inputs now contain more than one word):

{the cat sat} ([the sat], cat),
{cat sat on} ([cat on], the),
{sat on the} ([sat the], on)

Of course, as is explained in the tutorial, the words themselves aren’t actually used, but rather their (integer) index into the vocabulary (dictionary) for the task.

CBOW Input: Mean of Context Words Embeddings

In the skip-gram model there is just a single word as the input, and this word’s embedding is looked up, and passed to the predictor.

In the CBOW, since there’s more than one word in the context we just take the mean (average) of the embeddings for all context words.

Files

word2vec_fns.py	code for word2vec implementation.
word2vec_cbow.py	code to train word2vec model.
imdb_sentiment_data.py	helper functions for loading the sentiment data, used by word2vec_cbow.
plot_embeddings.py	to visualise embeddings.
reviews.tar.gz	25000 plain text movie reviews, split into positive and negative sets.
CBOW_Embeddings.npy	The results fitted onto the model trained.

Dataset
The training dataset contains a series of movie reviews scraped from the IMBD website. There are no more than 30 reviews for any one specific movie. You have been provided with a tarball (reviews.tar.gz) that contains two folders; "pos" and "neg". It contains the unchanged reviews in plain text form. Each folder contains 12500 positive and negative reviews respectively. Each review is contained in the first line of its associated text file, with no line breaks.