CS4740/5740 Introduction to NLP
Fall 2016
Word Clusters and Word Vectors

Due via CMS **and** Gradescope by Friday, 12/02 11:59pm

1 Overview

Word clusters and word vectors are common tools in NLP to overcome data sparsity problems in most statistical models. They place similar [1] words together in same clusters or nearby points in a vector space. In the more recent neural-network-based NLP systems, word vectors (or, the more popular name, "word embeddings") are playing increasingly important roles.

In this project, you are asked to improve any one of your previous projects with **either** word clusters **or** word vectors.

2 Grouping

We recommend groups of up to 4 people.

3 Project Description

In this project, you will mainly use pre-trained word clusters or word vectors from external resources (some helpful links can be found in the final section), and optionally train your own. Different packages may use different formats, and you should consult their corresponding documentations.

Before actual applications, you should start from simple observations. You need not to be an expert of how these clusters or vectors are derived, but it will be helpful to have a general understanding of their properties. In general, you will gain the intuition of what kind of "similarities" they are capturing.

- For word clusters, you may look at how words are grouped together. Some resources provide word clusters of different size, you may compare and experiment with different cluster sizes.

[1] Different algorithms have different definitions of being 'similar'. One of the most common assumptions made in word clustering and word embedding algorithms is that words appearing in similar contexts will get treated similarly.

- For word vectors, you may explore the vector space. For example, you may examine the nearest neighbors of some given words and analyze how similar words are arranged in the vector space.

All of the previous systems you were asked to build had data sparsity issues. You are asked to choose from any one of the previous projects, identify the sparsity issue it faces, and overcome the problem with the use of word clusters or word vectors. As some ideas to start, you may consider to:

- Include them as extra features (e.g. in CRF models for Project 2, or in the candidate ranking functions you designed for Project 3).
- Modify your previous models (e.g. in HMM and language models, you may propose a way to calculate emission/n-gram probabilities using word clusters). In this case, you need to justify the modified model and put down equations for calculation if necessary. (e.g. You need to show that the modified models are still producing well-defined probabilities.)

Optionally, if you get interested in how to train these word clusters or word vectors from corpora of your choice, you are encouraged to use existing software packages to train your own. You may further compare them with the pre-trained ones.

3.1 Summary

In summary, you are asked to:

- Download some off-the-shelf pre-trained word clusters or word vectors.
- Give some basic observations.
- Identity parts in any one of your previous projects where word clusters or word vectors will help. If you are proposing an improved model using these extra resources, write down any necessary equations.
- Modify code from your previous project, perform experiments, and analyze the results.
- (Optional) Train your own word clusters or word vectors with any existing package. Compare them with the pre-trained ones.

4 What to Turn in

- (via CMS) **Code of your system**. Do not submit word clusters or vectors. Instead, include URLs or scripts to obtain them.
- (via CMS **and** Gradescope) **Report** (maximum 6 pages, we may deduct points if you exceed this limit).

5 Rubric

Maximum score for this project is 50. You may get additional 5 points as bonus.

- (2 pts) Names, netIDs and contribution of every member in the group
- (18 pts) Code and implementation (you may reuse code from previous projects). You should highlight the changes you made in a readme file or in the report.
- (30 pts) Report (see below).
- (5 pts) Bonus for training your own word clusters or word vectors. To get full bonus points, you need to provide detailed analysis and comparison with the pre-trained ones.

5.1 Rubric for Report

	Excellent	Good	Fair
Basic observation and analysis (9 pts)	(9 pts) Detailed analysis of word clusters or word vectors and insightful observations.	(6 pts) Illustrative case analysis and adequate understanding of the cluster structure or vector space.	(3 pts) Some analysis and vague intuition about the "similarities" in the clusters or vectors.
Identification of the sparsity problem (3 pts)	(3 pts) Clear identification and explanation accompanied with illustrative examples.	(2 pts) General description with some explanation.	(1 pts) Only identifies minor issues.
Improvement over previous systems (6 pts)	(6 pts) Detailed justification for all design choices. Equations for modified models work out correctly.	(4 pts) Clear about the changes to be made, with adequate justification.	(2 pts) Shows effort to improve the previous systems.
Experiments and analysis (12 pts)	(12 pts) Considers different variations with detailed analysis. Conclusion shows understanding of word clusters or word vectors.	(8 pts) Clear about what to compare and adequate analysis of the results.	(4 pts) Experiments with the improved systems and some analysis.

6 Useful Links

- Pre-trained Brown clusters: `https://github.com/rug-compling/dep-brown-data` and `http://www.derczynski.com/sheffield/brown-tuning/#relatedlinks`

- Twitter word clusters: `http://www.cs.cmu.edu/~ark/TweetNLP/#resources`
- Word2vec word vectors (software and pre-trained vectors): `https://code.google.com/archive/p/word2vec/`
- Gensim package for using and training word2vec vectors in python: `https://radimrehurek.com/gensim/models/word2vec.html`
- Tensorflow tutorial for using word2vec (tensorflow is a popular neural network library): `https://www.tensorflow.org/versions/master/tutorials/word2vec/index.html`
- GloVe word vectors (software and pre-trained vectors): `http://nlp.stanford.edu/projects/glove/`