Assignment 3	LINGI2263	Dest. : Students
April 2012		Authors : A. Panchenko, A. Dessy

LINGI2263 – Computational Linguistics

Assignment 3 : Semantic Word Clustering

In this assignment, you will be asked to experiment a method for clustering words according to their meaning. This task is an application of lexical semantics that has become a major research area within computational linguistics.

1 Clustering Words with Similar Meaning

Your aim is to cluster input set of words $W = \{w_1, \ldots, w_n\}$ into disjoint groups containing words sharing similar meaning $C = \{C_1, \ldots, C_k\}$ (C form a partition of W). In the context of this work it is assumed that there is a semantic affinity between two words if :

- they are related with one of the following types of semantic relations : *hyponymy*, *hyperonymy*, *synonymy*, *association*, *meronymy*, *co-hyperonymy*. For instance, we can observe that $\langle w_i, hyponym, w_j \rangle = \langle alligator, hyponym, predator \rangle$.
- they are topically related. For example $C_i = \{w_1, w_2, w_3, w_4, w_5\} = \{baseball, game, football, pitch, hit\}$ would be a cluster related to sport.

If two terms w_i, w_j meet one of the following criteria then they should belong to the same cluster C_l, otherwise they should belong to different clusters. Your goal will be to implement a baseline word clustering method and investigate how well it completes this task.

2 Method

The semantic word clustering method, which you are going to implement, stems from the Lesk algorithm [2] as well as many other works in computational linguistics which exploit the following idea : "semantically similar words have similar definitions". In this work you are going to deal with definitions of words automatically extracted from Wikipedia and Wiktionary. The advantages of using these data with respect to the traditional dictionaries are the following : open access, huge coverage of the vocabulary in tens of languages, the resource is constantly updated by the community.

Each definition is a pair $\langle w, d \rangle$, where $w \in W$ is a single English word, and $d \in D$ is a definition of the word. This definition is a concatenation of the first paragraph of the Wikipedia article with title w and all entries in Wiktionary for w. The following example shows a definition of the term $w = alligator$:

alligator ; A large amphibious reptile with sharp teeth and very strong jaws related to the crocodile and native to the Americas and China. Informal short form : gator

The main step of the semantic word clustering method are the following :

1. Lemmatization and part-of-speech tagging of the definitions.
2. Filtering stop words and words with a stop part-of-speech from the definitions.

3. Representing each term $w \in W$ as an n-dimensional TF-IDF vector $\mathbf{v}$ [3], where n is the size of the vocabulary V. The ith element of this vector is defined as :

$$v_i = \frac{f_{id}}{\sum_i f_{id}} \times log \frac{|D|}{|d \in D : w \in d|},$$

where f_{id} is frequency of word $w_i \in V$ in definition d, $|D|$ is the number of definitions, $|d \in D : w \in d|$ is the number of definitions where the term w appears. The vector $\mathbf{v}$ is usually normalized as a unit vector.

4. Clustering words represented as vectors with the spherical k-Means algorithm [4]. The main difference of the spherical k-means from the original algorithm is use of the cosine as a similarity measure between words :

$$sim(w_i, w_j) = \frac{\mathbf{v}_i^T \mathbf{v}_j}{||\mathbf{v}_i|| \cdot ||\mathbf{v}_j||}.$$

We are going to adopt this algorithm because cosine is a more suitable similarity measure than Euclidean distance for text data (why ?). The algorithm is presented below [1]

Algorithm 1: Spherical k-means algorithm

Input: A set of $N = |W|$ unit-length bag-of-words vectors $X = \{\mathbf{v}_1, \ldots, \mathbf{v}_N\}$ each representing a word $W = \{w_1, \ldots, w_N\}$ and the number of clusters K

Output: A partition of the words by the cluster identity vector $Y = \{y_1, \ldots, y_N\}, y_n \in \{1, \ldots, K\}$

1 Initialization : initialize the unit-length cluster centroid vectors $\{\mu_i, \ldots, \mu_K\}$;
2 Data assignment : for each data vector $\mathbf{v}_n$, set $y_n = \arg\max_k \mathbf{v}_n^T \mu_k$;
3 Centroid estimation : for cluster k, let $X_k = \{\mathbf{v}_n | y_n = k\}$, the centroid is estimated as $\mu_k = \sum_{\mathbf{v} \in X_k} \frac{\mathbf{v}}{||\sum_{\mathbf{v} \in X_n} \mathbf{v}||}$;
4 Stop if Y does not change, otherwise go to Step 2 ;
5 **return** Y ;

Your solution can be implemented in Java or in any scripting language of your choice (Python, Ruby, Perl, Matlab, R). You should use an existing part-of-speech tagger and lemmatizer such as TreeTagger [2]. You can integrate a library implementing the clustering algorithm or implement it yourself.

3 Dataset

The dataset encompasses 6914 single English words and their definitions extracted from DBpedia [3] and Wiktionary [4]. It is available from the course website :

`http://icampus.uclouvain.be/courses/INGI2263/document/data/WikiDefs/defs.zip`.

1. This algorithm description was adopted from [4]. Would you need more information about the algorithm please refer to this article and [1].
2. `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/`
3. `http://dbpedia.org/`
4. `http://www.wiktionary.org/`

4 Products

Turn in your assignment as an archive containing the code and a report (in PDF format) in due time. Your script command line interface should be similar to :

> python semantic_words_clustering.py definitions_file clusters_file

where definitions_file is the path of the input file with a list of words and their definitions and clusters_file is the path of the file where the script puts the clustering results. Each line of the output file should list the words in a cluster C_k in the following format :

$k; w_1; w_2; \ldots; w_m,$

where k is the identifier of cluster C_k composed of m words : $\{w_1, \ldots, w_m\}$. Thus, a line of an output file may look as following :

44 baseball ;football ;ball ;game ;pitch ;hit

Your report (5 pages maximum) should contain two parts :

Global architecture

Describe the architecture of your program and its different components, as well as the preprocessing applied to the definitions. Please justify your choices regarding this architecture and keep in mind the reproducibility of your solution.

A discussion on the results

1. Run the semantic clustering with different values of K. Look through the results. Does small values of K provide better results than large values at the first glance ? Choose two values of K which provide the best results in your opinion and attach the corresponding files to the report.
2. How can we evaluate the quality of the clustering in a more automated way ? What kind of resources will you need to implement this evaluation ?
3. Can you possibly think of ways to enhance your results ? What may be the reasons of the errors in your results ?

Références

[1] I.S. Dhillon and D.S. Modha. Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1) :143–175, 2001.

[2] M. Lesk. Automatic sense disambiguation using machine readable dictionaries : how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, pages 24–26. ACM, 1986.

[3] C.D. Manning, P. Raghavan, and H. Schutze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[4] S. Zhong. Efficient online spherical k-means clustering. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 5, pages 3180–3185. Ieee, 2005.