Dictionary Definitions: The likes and the unlikes

Anagha Kulkarni
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
anaghak@cs.cmu.edu

Abstract

In this work we try to approach the task of grouping word definitions from English as Second Language dictionaries based on the similarity of their meanings using unsupervised machine learning algorithms namely spectral clustering, hierarchical clustering and K-means. It is encouraging to see that simple lexical features along with a combination of the above unsupervised clustering methods can deal with the data which consists of very short sentences (definitions) and very few data-points (definitions) per class/cluster, with high accuracy.

1 Introduction

The REAP project[1] aims at providing assistance to ESL (English as Second Language) students to improve their vocabulary. For every grade level the human teacher has a list of new words (focus-words) that he/she would like the students to learn. Currently, it is this set of focus-words that REAP tries to assist the students with. The approach is to pre-test a student to determine his current vocabulary and then retrieve passages from World Wide Web (WWW) that contain 3 or 4 focus-words. Each of the retrieved passages goes through automatic filters, like, reading level, quality and length before it can reach the student. However, student is the first person to look at the document, that is, there is no background human/teacher inspection for these passages.

A machine readable version of Cambridge Advanced Learners Dictionary (CALD) is integrated in REAP which the students can use while they read a given passage, to lookup the focus or non-focus words. As a result, a student has two knowledge sources to learn the meaning of the new word – the context of the new word and its definition from the dictionary. For a native speaker of English typically these two resources together are sufficient to understand the meaning of a new word precisely. However, for a non-native speaker multiple factors make these two resources not as

[1] http://reap.cs.cmu.edu/

effective as they are for a native speaker. The one that we are interested here is the case where the dictionary entry shows multiple definitions for the looked-up word. Many words in English have more than one sense and thus have multiple definition entries in a dictionary (assuming the dictionary has a good coverage of words and word-senses). Few such ambiguous word-senses can be disambiguated based on their part-of-speech category (noun, verb, adjective, and adverb) in a given sentence. For example: the "endowment" sense of the word "grant" can be disambiguated from the "to give" sense by recognizing that the first sense will be used when the word "grant" is in the noun position while the second sense in the verb position in a sentence/phrase. However, many times multiple definitions with different senses (homonyms) exist under the same pos categories. For example: Under the verb pos category of the word "grant" there still exist two distinct definitions – "to give" and "to assume". A related phenomenon that we often observe is that for more than negligible words there exists multiple definitions with very similar meaning (polysemy). Such polysemy can potentially confuse a non-native language learner unless they are grouped together. For example: CALD gives two definitions for the word "bias" under the noun pos category – "a tendency to support or oppose a particular person or thing in an unfair way by allowing personal opinions to influence your judgment" and "a preference towards a particular subject or thing", another example could be the two definitions for word "accumulate", "to collect a large number of things over a long period of time" and "to gradually increase in number or amount", both of which convey the same or highly similar meaning. At this point, we would like to state explicitly that such grouping of definitions based on the similarity of the meaning being conveyed is a highly subjective task and different schools of thought might favor different groupings. The philosophy that we have used while annotating the definitions was to place the word along with the definition under consideration within the context of a set of different sentences and to see which definitions were acceptable in a same subset of sentences and were not. We also had to careful to limit the amount of background knowledge about English that we assumed during the annotation.

2 Problem definition

Given the above background and the motivation, the problem we consider here is that of learning to cluster/partition a set of given definitions into groups such that polysemous definitions are grouped together while the homonym definitions are separated out.

Secondly we have observed that instead of using just one dictionary, a combination of two or more dictionaries usually works better to cover all the possible senses of a word in use. Thus, in near future REAP will integrate two dictionaries, CALD and Longman Dictionary of contemporary English (LDCE). Combining dictionaries, along with new definitions also brings redundant definitions which we would certainly want to eliminate. As one can see this addition further increases the difficulty but at the same time the necessity of the task at hand.

3 Literature Review

In [1] the author uses machine readable dictionaries to perform automatic word sense discrimination. More specifically, given a sentence containing an ambiguous word (target-word), the aim is to associate the most appropriate dictionary definition of the word based on the word's intended meaning in the sentence. The proposed approach is to find the word-overlap between the definitions of the target-word and the definitions of the words in its immediate vicinity. For example, to associate the

correct meaning of the word "cone" in a sentence where the word "pine" precedes it the word-overlap between the definitions of "cone" and the definitions of "pine" will be computed.

[2] take a completely unsupervised approach to identify and cluster words or instances, where an instance can virtually be any unit of text. The approach is based on the principle of contextual similarity according to which the ambiguity about the semantics of a word can be resolved by looking at its surrounding context. More specifically, the authors use lexical features (unigrams: single words, bigrams: ordered word-pair, co-occurrences: unordered word-pair) and transform each textual instance into a vector representation using a direct first order or in-direct second order vector representation. The instance vectors can then be clustered using various different clustering algorithms supported by the clustering suite CLUTO[2].

4 Data Description

The dataset consists of 383 definitions for 80 words from two ESL dictionaries: CALD and LDCE. The 383 definitions have been manually grouped into 192 groups/classes. 59 classes have 1 data-point, 90 classes have 2 data-points, 33 classes have 3 data-points, 6 classes have 4 data-points, 3 classes have 5 data-points and 1 class has 6 data-points! On an average each definition consists of 12 words.

Example: Definitions for the word "grant" from CALD and LDCE:

1. to give or allow someone something, usually in an official way
2. to accept that something is true, often before expressing an opposite opinion
3. to give someone something or allow them to have something that they have asked for
4. to admit that something is true although it does not make much difference to your opinion
5. to believe that something is true without making sure
6. to expect that someone or something will always be there when you need them and never think how important or useful they are

5 Proposed Method

Given the nature of the dataset, that is, short sentences and highly sparse classes, training a classifier had to be ruled out. Instead, we have experimented with a couple of unsupervised clustering algorithms and a range of feature types, namely:

1. Raw word-overlap with and without stopwords
2. Normalized word overlap with and without stopwords
3. Cosine similarity with and without stopwords
4. Raw word-overlap with a range of term frequency-inverse document frequency (tf-idf) cutoffs.
5. Normalized word-overlap with a range of tf-idf cutoffs.
6. Cosine similarity with a range of tf-idf cutoffs.

Each of the above feature type operates on a pair of definitions and thus leads to a symmetric adjacency/affinity matrix. It is important to note that every definition for

[2] http://glaros.dtc.umn.edu/gkhome/views/cluto/

a word is compared (i.e. feature scores are computed) only with the other definitions of the same word, in other words, definitions are not compared across words.

The raw word-overlap is simply the number of words common to both the definitions of the pair under consideration. Please note that this overlap is based only on the words as they appear in the definitions, that is, their morphological roots were not consulted to decide an overlap. When using lexical features, i.e., not using any syntactic information, most often than not, including the function or closed-class words like articles and prepositions does not help, in-fact they can potentially mislead features such as the overlap scores. For example, in the above example for word "grant", the word "to" occurs in all the six definitions and thus a word overlap of 1 is evident among all the 15 possible pairs of the six definitions. To avoid this, we also experiment with function words free definitions. That is, we remove all the function words from the definitions being compared before we measure the word-overlap between them. The stop-list that we use is fairly conservative keeping in mind the size of the definitions. It consists of the following function words: articles (*a, an, the*) and prepositions (*of, to, in, for, on, with, as, by, at, from*) and an auxiliary verb *be*.

The normalized word-overlap feature type bounds the overlap scores to the range of [0,1] by scaling with respect to the definition length (in words). Doing so makes it easily possible to compare two overlap scores. For example, it might not be obvious that an overlap of 4 words between definitions with lengths 10 and 12 is smaller than an overlap of 3 words between definitions of lengths 7 each. The actual formulation used for computing the normalized word-overlap (*nwo*) between definitions d_a and d_b is:

$$nwo(d_a, d_b) = \frac{2 * \frac{rwo(d_a, d_b)}{|d_a|} * \frac{rwo(d_a, d_b)}{|d_b|}}{\left(\frac{rwo(d_a, d_b)}{|d_a|} + \frac{rwo(d_a, d_b)}{|d_b|} \right)}$$

Where, *rwo* stands for the raw word overlap count between definitions d_a and d_b and $|d_x|$ gives the length of the definition x in words.

For the cosine similarity score each definition is represented as a vector of "bag-of-words" features selected from the two definitions under consideration. And then cosine similarity is computed between the two vectors. Here too the "with" and "without" stopwords options are used.

Term frequency inverse document frequency (tf-idf) is a commonly used feature/term selection measure, especially in Information Retrieval. The aim is to assign each term in a given document a score which represents the importance of the term to the document. The higher this score of a term the more pertinent it is to the document. The term frequency part is computed by counting the number of times the term occurs in the given document and normalizing it with the length of the document and the inverse document frequency part is computed by taking the log of the ratio of total number of documents and the number of documents in which the term occurs. The overall tf-idf score is computed by simply multiplying the term frequency and the document frequency scores. Thus this measure is based on the premise that the frequency information about the term can be used to rate the relevance of the term to the document. For the problem at hand, using individual definitions as documents does not work well because of the short length of the documents, that is, the definitions. Therefore we have combined definitions annotated with the same class to form a document. We then apply tf-idf measure to

rank the terms/features of such documents. We have experimented with different percentage cut-offs of the ranked feature list. It should be noted here that this is a supervised feature selection and thus is only used for the purpose of comparison.

In the feature types 4, 5 and 6 above, instead of using all the unique words or all the unique content words from the pair of definitions, to represent the definitions, we use words selected based on their tf-idf scores. More specifically, we choose top n% words from each document (i.e. group of definitions from the same class) and represent the definitions in terms of these features. Again, we do realize that such feature selection will not be possible in real application.

On the algorithms front, we have experimented with the following different combinations of K-means, Hierarchical and Spectral clustering:

1. K-means [3]
2. Spectral clustering (Ng et. al [5]) followed by K-means [3]
3. Spectral clustering (Ng et. al [5]) followed by Hierarchical clustering (Ward's algorithm [4])
4. Spectral clustering (Ng et. al [5]) followed by Spectral clustering (Ng et. al [5]) followed by K-means [3]
5. Spectral clustering (Ng et. al [5]) followed by Spectral clustering (Ng et. al [5]) followed by Hierarchical clustering (Ward's algorithm [4])

Each of the above algorithms is applied to the adjacency matrix created by each of the above described feature type separately to compare the effectiveness of each of the feature type and the algorithm. The number of clusters were set manually in all these experiments. Trying to automate the choice of number of clusters will be a part of the future work.

K-means algorithm [3] starts with k random cluster means, where k is specified by the user and all the data-points (here definitions) are assigned to the closest cluster mean. The definition of closeness used here is cosine similarity. Next, each of the data-point is re-assigned to a cluster if doing so improves the overall similarity score. The cluster mean is recomputed every time a new data-point is assigned to that cluster. We repeat the re-assigning process 100 times or until no more re-assignments occur, to avoid local optimums. The complete clustering of data is repeated 5 times, every time starting with a different set of k random cluster means to avoid errors introduced due to non-ideal initial cluster means. The best solution of the 5 clustering solutions is chosen.

The Ward's algorithm [4] starts with each data-point (definition) in its own cluster and at every step merges a pair of clusters that minimizes the loss in information, i.e., minimizes the change in the objective function value, which is squared sum of squares here.

The spectral clustering algorithm proposed by Ng et. al [5] transforms the higher dimensional feature vectors (d) to a lower spectral dimension (k). More specifically, given a similarity/affinity matrix (W) of d dimensions, a diagonal matrix (D), which is sum of every row of the affinity matrix placed along the diagonal, is computed. A Laplacian matrix ($L = D^{-1/2}\mathrm{W}\ D^{-1/2}$) is computed and its eigen-components are computed. The eigenvectors corresponding to the top k eigenvalues are selected to be represented as columns of a new matrix (X) and then the rows of X are normalized to have unit length. The rows of this normalized matrix are now clustered as one would cluster the original data-points; however, the dimension of the new vectors is k and not d. Hence forth we will refer to the above method as

NJW. In the experiments performed here with NJW we have tried K-means and Ward to cluster the data-points in the reduced spectral space. We have also tried re-applying NJW in the spectral dimension with the intention of investigating if applying NJW in the reduced spectral space further adds any value. We have used *sigma=0.2* in all the experiments. The *sigma* value is used in the similarity computation, while generating the affinity matrix. This experimental setup is based on [6].

6 Results and Discussion

Table 1 presents the results of the five clustering algorithms (along the row) when used with different variants on the raw word-overlap (rwo) feature type. The second column records the results when using all the unique words in the definition pair, the third records the results when using unique content word, fourth when only top 10% words from each class's tf-idf ranked list were used and so on. The results are in terms of the clustering error which is simply a ratio of number of misclassified definitions and the total number of definitions (383). It is interesting to note that when using the raw word-overlap feature type spectral clustering does not add any value, in fact directly using the K-means algorithm is most effective. It is encouraging to see that the simple stop-list is most effective with this feature type.

Table 1: Results (clustering error) when using raw word-overlap feature

Algorithm	**rwo w/ stop-words**	**rwo w/o stop-words**	**rwo w/ tfidf = 10%**	**rwo w/ tfidf = 30%**	**rwo w/ tfidf = 50%**	**rwo w/ tfidf = 70%**
K-means	0.289817	**0.245431**	0.467363	0.477807	0.454308	0.480418
NJW-Kmeans	0.315927	0.331593	0.37859	0.357702	0.347258	0.342037
NJW-Ward	0.310705	0.32376	0.391645	0.368146	0.35248	0.349869
NJW-NJW-Kmeans	0.284595	0.29765	0.35248	0.321149	0.334204	0.328982
NJW-NJW-Ward	0.289817	0.308094	0.349869	0.32376	0.326371	0.331593

Table 2 shows the results when using normalized word-overlap feature type. The table format is similar to Table 1. We can see that normalizing the word-overlap score buys a lot for spectral clustering, especially when it is followed by the hierarchical clustering, Ward's method. It is important to note that an additional second stage of spectral clustering, even if followed by Ward's algorithm, does worst than just one phase of spectral followed by Ward's method.

Table 2: Results (clustering error) when using normalized word-overlap feature

Algorithm	**nwo w/ stop-words**	**nwo w/o stop-words**	**nwo w/ tfidf = 10%**	**nwo w/ tfidf = 30%**	**nwo w/ tfidf = 50%**	**nwo w/ tfidf = 70%**
K-means	0.305483	0.242820	0.454308	0.464752	0.462141	0.467363
NJW-Kmeans	0.232376	0.201044	0.224543	0.224543	0.227154	0.224543
NJW-Ward	0.229765	**0.198433**	0.245431	0.227154	0.234987	0.237598
NJW-NJW-Kmeans	0.250653	0.214099	0.245431	0.237598	0.253264	0.263708
NJW-NJW-Ward	0.253264	0.21671	0.248042	0.234987	0.258486	0.263708

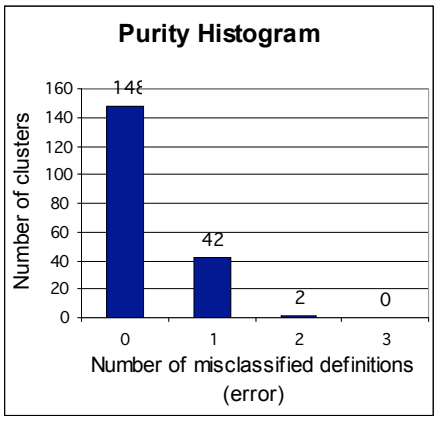
Table 3 shows the results when using cosine similarity feature type. The table format is similar to Table 1. This feature type gives a further improvement, again with spectral and hierarchical algorithms. Overall it is evident that applying spectral clustering and thus re-representing the definitions in the reduced space in terms of their eigenvectors does help. In the original space the data is not normally distributed and thus K-means struggles. However the transformation performed by NJW does significantly help K-means performance.

Table 3: Results (clustering error) when using cosine similarity feature

Algorithm	**cos w/ stop-words**	**cos w/o stop-words**	**cos w/ tfidf = 10%**	**cos w/ tfidf = 30%**	**cos w/ tfidf = 50%**	**cos w/ tfidf = 70%**
K-means	0.289817	0.258486	0.258486	0.266319	0.258486	0.263708
NJW-Kmeans	0.245431	0.193211	0.203655	0.219321	0.21671	0.229765
NJW-Ward	0.240209	**0.18799**	0.216710	0.214099	0.221932	0.229765
NJW-NJW-Kmeans	0.24282	0.211488	0.211488	0.232376	0.24282	0.245431
NJW-NJW-Ward	0.248042	0.211488	0.229765	0.232376	0.24282	0.24282

Evaluating clustering performance is almost always tricky because to start with, clustering is a very subjective task and thus any gold standard is unlikely to be universally accepted, secondly the evaluation metric(s) to be used should depend upon the task. Following is an in-depth analysis of the best case: NJW-Ward and cosine without stopwords, which we will hence forth refer to as NWC.

The metric used for computing the clustering error reported in the above tables penalizes cases where members of originally one cluster were split by the clustering algorithm into 2 or more pure clusters. Instead, if we measure the performance of the clustering algorithm in terms of the impurity of the generated clusters then for the best case NWC, the error drops down to 0.1201 (46/383). In more details, the numerator in the previous calculation (46) is the number of definitions that made an otherwise pure cluster impure, i.e., all the definitions that do not belong to the majority group within their cluster. The histogram of purity of clusters is shown in Figure 1 below. This figure shows that 148 clusters were totally pure, i.e., had zero misclassified definition, 42 clusters had one misclassified definition, 2 clusters had two misclassified definitions and none of the clusters had more than two misclassified definition. In other words, 42 * 1 definitions + 2 * 2 definitions were misclassified. To get the complete picture Figure 2 provides the histogram of cluster-size of both, the clustering solution given by NWC and the gold standard. As we can see from the plot, the proposed clustering solution by NWC comes quite close to the gold standard cluster-size wise too. However NWC seems to be struggling with cluster sizes greater than four. Although, for the task of clustering definitions the size of a cluster would typical not exceed five or six definitions, trying to find a solution for this problem will be a part of the future work. We can also see that NWC has confused a few (17 definitions) of the single element clusters by combining them into larger clusters. This is another direction of the future work – to find feature types which will be able to capture better discriminating features to avoid such groupings. We also plan to look at options which might help us enrich or expand our terse definitions and thus help us build richer definition representation.



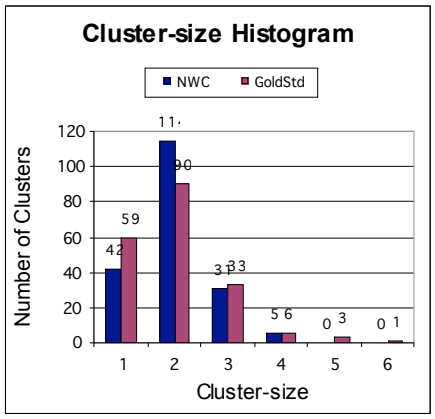


Figure 1: Purity Histogram for NWC **Figure 2: Cluster-size Histogram**

7 Conclusion

This work shows that spectral clustering, more specifically [5], when followed by hierarchical clustering, more specifically [4], can be successfully used to cluster non-Gaussian data with large number of classes and very few members per class. Similarity metrics such as cosine along with purely lexical features like single words (unigrams) when filtered with a simple stop-list can be effectively used to capture the (dis)similarity between definitions.

References

[1] Lesk M. E. (1986) Automatic Sense Disambiguation using Machine Readable Dictionaries: How to tell a pine cone from an Ice Cream Cone. SIGDOC, Toronto, Canada.

[2] Pedersen T. and Kulkarni A. (2005) Identifying Similar Words and Contexts in Natural Language with SenseClusters, ACL (Intelligent Systems Demo), Ann Arbor, MI.

[3] MacQueen J. B. (1967) Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press.

[4] Ward J. H. (1963) Hierarchical grouping to optimize an objective function. Journal of American Statistical Association, 58(301), 236-244.

[5] Ng A., Jordan M., and Weiss Y. (2001) On spectral clustering: Analysis and an algorithm. Advances in Neural Information Processing Systems.

[6] Verma D. and Meila M. (2003) A comparison of spectral clustering algorithms. Technical Report 03-05-01, Department of Computer Science and Engineering, University of Washington.