Clustering of Co-Occurring Neighboring Unambiguous Terms (COCONUT)

Anouk Visser anouk.visser@student.uva.nl
Rémi de Zoeten remi.de.z@gmail.com
Cristina Gârbacea 1st1na.garbacea@gmail.com

Abstract

Let's decide on the abstract once we finish the body of the text.

1 Introduction

The introduction will be here.

2 Related Work

Recently (Mikolov et al., 2013a) found that continuous space word representations capture syntactic and semantic regularities. For example they find that *queen* $-$ *king* $+$ *man* $\approx$ *woman*. The authors use these linguistics regularities to answer a set of analogy questions in the form of 'a is to b as c is to ...'. In addition to this, the linguistic regularities can also be used to compute similarity of the relations of pairs of words $a : b$ and $c : d$.

The linguistic regularities in continuous space word representations can be identified by using a vector offset method based on cosine similarity. A recurrent neural network language model is used to obtain the continuous space word representations. The RNN is trained using backpropagation to maximize data likelihood and consists of one input layer that accepts one word at a time encoded using *1*-of-*N* encoding scheme, an output layer which outputs a probability distribution over possible words and a hidden layer with recurrent connections that keeps track of the sentence history. The embedding vectors x_a, x_b, x_c are used to determine the word which is assumed to be the best answer to a question, $y = x_b - x_a + x_c$, or in case there is no word in space at this position, the word having the greatest cosine similarity with y. In the case of semantic evaluation where d is given, computing $cos(x_b - x_a + x_c, x_d)$ determines the measure of relational similarity between the prototypical and target word pairs.

There is a huge potential for using linguistic regularities in unsupervised language learning applications. An example application is presented by (Mikolov et al., 2013b), where a method is proposed to exploit similarities among languages for machine translation. For machine translation some sense of a dictionary or phrase table is required. However, these are not always available, or are incomplete. By using two monolingual corpora a model can be trained and linguistic regularities can be learned for the two different languages. If the dictionary entry for 'queen' is missing, but, for example, the entry for 'king' is available, we can find the translation for 'queen' by using the vector offset method as described above.

3 Word disambiguation by word-level co-occurrence clustering

One potential problem when answering analogy questions is that words can be ambiguous. When the question is *Apple is to XXX what XXX is to ...* then the word apple refers to a fruit. Another question could be *Apple is to XXX what XXX is to ...* where the company apple is implied. Clearly, these are two very distinct entities, but the approach that Mikolov et al. presented does not differentiate. It would be appropriate to disambiguate between the various senses of the word apple, before answering the question. One approach is to cluster different meanings of a word based on the words that it co-occurs with in the corpus. This is done by generating a co-occurs vector for every time the word is observed in the corpus. The co-occurrence vector is derived by observing the context of a word. In our experiments the frequency of the words that fall within a window of 5 words from the word that is being observed are encoded into the vector. This means that each vector is a sparse vector with the length of the vocabulary size, but can be encoded with at most 10 terms. It would also be possible to use a soft-max or gaussian measure to weigh each word based on its distance, but we have weighted each of the 10 words in the 5 word window equally.

These co-occurrence vectors are then clustered using k-means clustering. It is possible (even likely) that two co-occurrence vectors have no word in common, but still end up in the same cluster. For example, in the case of apple the words {*technology, iphone, company, revenue*} might be in the same cluster. Given the co-occurrence vectors, 1:{*technology, iphone*}, 2:{*iphone, revenue*}, 3:{*technology, company*} and 4:{*company, revenue*} then 1 and 4 have nothing in common, but can still be bound together by 2 and 3. It should be noted that extracting all co-occurrence vectors from a corpus can can require a significant amount of memory, even when using sparse-vector encoding. However, it is possible to have a fine-tunable tradeoff between memory requirements and the number of loops over the corpus (which is more cpu-intensive), by only recording a specific subset of the vocabulary on each iteration.

4 COCONUT

For learning the word representations (Mikolov et al., 2013a) train an RNN with co-occurrence vectors of words. Instead of representing words by just one co-occurrence vector, we propose to train the model with multiple co-occurrence vectors for ambiguous words. The meaning of the word 'apple' can be determined by looking at its surrounding words, which could be: technology, iPhone, company for 'Apple', the company or: fruit, orchard, pie for 'apple' the fruit. COCONUT assumes that the meaning of a word is highly dependent on the words that accompany it and that the co-occurring words that define one meaning of 'apple' are more likely to co-occur with each other than two words that define two different meanings of apple ('iPhone' and 'technology' are more likely to occur together than 'iPhone' and 'orchard'). COCONUT will attempt to split the co-occurrence vector for 'apple' into two co-occurrence vectors, one containing 'iPhone', 'technology' and 'company', the other containing 'fruit', 'orchard' and 'pie'.

4.1 Co-Occurrence Vectors

We construct the co-occurrence vector for word A by computing the relatedness of word A with every other word in the vocabulary. We use the same function for relatedness as (Guthrie et al., 1991):

$$r(x, y) = \frac{f_x y}{f_x + f_y - f_x y}$$

where $f_x y$ denotes the frequency of x and y occurring together and f_x and f_y denote the frequency of x, respectively y.

4.2 Clustering

To find the two senses of a word, we apply k-means clustering to the co-occurrence vectors of the co-occurring words. COCONUT assumes that the words assigned to each cluster represent a different meaning of a word. Words that are not closely related to A do not contribute to either one of the meanings. Therefore, we will not use the co-occurrence vectors of all co-occurring words, but only those from the words that are closely related. Building a good decision process for defining when a word is closely related to another word is beyond the scope of this project and will most likely not necessarily lead to significant performance improvements. Therefore, we have decided to discard the words that have a relatedness score with A that falls in the bottom 50% of all relatedness-scores. Let the set of words that remains be called C. We can use the co-occurrence vectors of the words in C to find clusters, but these vectors will contain a lot of words that are not in C, do not occur together with A or do occur with A but not in C. We are only interested in finding clusters representing the different meanings of word A, therefore we will only use the co-occurring words in the vectors of C that are present in C.

5 Corpus annotation and question answering.

Once a set of ambiguous words and their representation has been identified a new corpus is generated wherein the ambiguous words are annotated with their meaning. This is done by looping over the words in the corpus and again extracting the context of the words that are ambiguous. This context is then matched with either of the clusters that were found for that particular word. If the context of a word is closest to the representation of cluster 1, the word will be annotated with a '_1' and if the word context corresponds more with the second meaning of the word the word is annotated with '_2'. Then, the process described in Mikolov et al. is repeated to get the word-vector representation of the words in the annotated corpus. Now the question triplet *a : b : c* can be translated into many interpretations, namely *a_0 : b_0 : c_0* , *a_0*

: b_0 : c_1, etc. If all three words are ambiguous then there are 8 possible interpretations. Of course, not all interpretations are sensible. If the question is *king : queen : man* then the interpretation of *queen* as a band is not sensible, but should be interpreted as 'queen as-in royalty'. To achieve this result we first ask the question *which a and which b are most similar?*. This is done by combining all *a* and all *b* and measuring their distance. It is assumed that royalty king and royalty queen are closer together than, for example, card-game king and the band queen. This way the question is reduced to two questions, where *c* is still ambiguous. Now both questions will be answered and an error $||\ e\ ||$ is determined for each answer, such that $b - a + c \equiv answer + e$. Finally we choose the answer with smallest error.

6 Latent Semantic Analysis

XXX I think this might be better in the related work section, also because we don't have measurements for this method XXX
Unsupervised word sense disambiguation approaches exploit the idea that similar senses of a word have similar neighboring words. They try to induce word senses from input text by clustering word co-occurrences, aiming to divide "the occurrences of a word into a number of classes by determining for any two occurrences whether they belong to the same sense or not" (Schutze et al., 1997).

7 Similarity Measures

XXX I don't think we should have a section on similarity measures. I think it would be good to mention it in the results section. XXX
Measuring the similarity between two vectors can be seen as an equivalent to measuring their distance. Inversion or subtraction can be easily applied to transform a measure of distance between vectors into a measure of similarity.

The most common way to measure similarity between two vectors is to compute the *cosine* of the angle between them as the inner product of the two vectors, after they have been normalized to unit length: $cos(x, y) = \frac{x \cdot y}{||x|| \cdot ||y||}$. Hence the length of the vectors is irrelevant. (Bullinaria et al., 1997) show that the cosine is highly reliable and performs the best, after having compared it with distance measures like Hellinger, Bhattacharya, and Kullback-Leibler. Other common geometric metrics frequently used in the vector space are represented by the Euclidean, Manhattan and Mahalanobis distance, Dice, Jaccard, Pearson and Spearman correlation coefficients.

The *Euclidean* distance between two points is defined as the length of the line connecting them. In the vector space, it is defined as $d(p, q) = d(q, p) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$. The smaller this distance the more similar the objects are. In a similar manner, the *Manhattan* distance is defined as the sum of the absolute differences of the coordinates of two given points as $d(p, q) = |\sum_{i=1}^{n}(p_i - q_i)|$. The *Mahalanobis* distance generalizes the standard Euclidean distance by modelling the relations of elements in different dimensions. Given two vectors x and y, their squared Mahalanobis distance is $d_A = (x - y)^T A(x - y)$, where A is a positive semidefinite matrix.

The *Pearson* correlation coefficient is defined in a similar manner with the Spearman correlation coefficient, with the mention that the last one is between the ranked variables: $\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 (y_i - \bar{y})^2}}$, where $\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$ and $\bar{y} = \frac{\sum_{i=1}^{n} y_i}{n}$. A value of 1 indicates total positive correlation, i.e. that a "best fit" line with a positive slope is generated which runs through all the datapoints, a value of 0 means no correlation and a value of -1 represents negative correlation between the two variables. The main advantage of this method over the Euclidean distance is that it is more robust against data which is not normalized.

8 Evaluation

We have evaluated the performance of COCONUT on a dataset containing X unique words, and has size X. Initially, we decided not to disambiguated the top X words, after extracting the two senses of the words and their distance, we discarded half of the disambiguated words, leaving us with X words that were disambiguated.

8.1 Empirical Evaluation

8.2 Quantitative Evaluation

9 Conclusion

References

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013a. Linguistic regularities in continuous space word representations. Proceedings of NAACL-HLT, 746–751

Tomas Mikolov, Quoc V. Le and Ilya Sutskever. 2013b. Exploiting Similarities among Languages for Machine Translation. arXiv preprint arXiv:1309.4168,

Joe A. Guthrie, Louise Guthrie, Yorick Wilks and Homa Aidinejad. 1991. Subject-dependent co-occurrence and word sense disambiguation. Proceedings of the 29th annual meeting on Association for Computational Linguistics, 146–152 Association for Computational Linguistics

John Bullinaria and John Levy. 1997. Extracting semantic representations from word co-occurrence statistics: A computational study. Behaviour Research Methods, 510–526

Schutze H. 1998. Automatic Word Sense Discrimination. Computational Linguistic, 97