Learning semantic vector representations

Amr Aly (1848399)

June 11, 2019

Abstract

We use automatically sense-annotated corpora to learn word sense embeddings by training a Word2Vec model and evaluate it on a word similarity task.

Data Preprocessing

We use three datasets for training our model, Eurosense, Semantically Enriched Wikipedia (SEW) and Train-O-Matic. Variations were constructed from these three to be experimented upon, however the variations occur only for the SEW dataset. Table 1 illustrates the different variations used.

We use all 1.9M sentences from Eurosense and all 100K sentences from Train-O-Matic. For SEW, first we use the first 4M sentence (max number that could fit into RAM), then we observe that a wikipedia article has many similarly annotated words, so we decide to extract the 4M by randomly sampling 20% of the sentences of each article. Next, we try to extract a large amount of sentences ($>$ 10M) from SEW by writing the sentences directly to a file instead of storing them into RAM.

Sense annotations are provided in BabelNet synsets, so we map them into WordNet synsets and discard any annotation that is not already present in this mapping (i.e. not in WordNet).

We preprocess any text by removing any punctuation characters and then remove extra space (consecutive spaces $>=$ 2) in order to keep the original structure and split tokens on spaces.

For Eurosense, we use the high precision version of the dataset, so if there exists overlapping mentions, we use the longest mention, since it is the most specific annotation.

Word2Vec

We use the gensim implementation of Word2Vec CBOW architecture, with embedding size of 400, 15 iterations and a windows of size 10 in order to capture more context. Due to the large number of vocabulary, we utilize two methods to reduce training time; first, we use the annotations only found in WordNet to reduce the dimensions of the embedding matrix, second, we apply hierarchical softmax.

The hierarchical softmax is an efficient approximation of the full softmax. Instead of evaluating V output nodes to obtain the probability distribution, it only needs to evaluate $\log_2 V$ nodes since it uses a binary tree to represent the output layer of V words.

The model outputs a file containing *only* the sense embeddings in standard Word2Vec format.

Word Similarity Task

We perform the word similarity task as required. We use both the cosine and weighted cosine similarity measure [1] and calculate the spearman correlation coefficient between the gold scores and the calculated scores. We notice from Table 2 that the weighted cosine measure gives better results.

Embeddings Analysis

We perform k Nearest Neighbor to analyze the output of the model. We also visualize the embeddings using PCA and t-SNE.

We notice that we achieve satisfactory correlation measure and good results in general. However, the model needs more examples to train. This is clear in the t-SNE plot Figure 2 where not all word clusters make sense. For example the cluster for the meaning *Seek* looks to make, however in the rest we may find a few relevant words in each cluster.

On the other hand, performing k Nearest Neighbor analysis (on a relatively easy example), Table 3 where we analyze two meanings of the word *Bank* {financial institution and river bank} produces meaningful results.

To conclude, giving the system more examples and time to train would produce better embeddings for the sense. However, with simple training we can notice satisfactory results of the system.

References

[1] Iacobacci, I., Pilehvar, M.T. and Navigli, R., 2015. Sensembed: Learning sense embeddings for word and relational similarity. https://www.aclweb.org/anthology/P15-1010

Label	Combination
D1	Eurosense + Train-O-Matic
D2	D1 + 4M of SEW
D3	D1 + 4M of SEW (randomly sampled)

Table 1: Datasets Variations.

Dataset	Cosine	Weighted Cosine
D1	0.18	0.20
D2	0.48	0.52
D3	0.45	0.48

Table 2: Word Similarity Results.

Bank (financial) 'bank_08420278n'	**Jaccard Similarity**	**River Bank** 'bank_09213565n'	**Jaccard Similarity**
financial_institution_08054721n	0.991	river_09411430n	0.954
investment_bank_10215953n	0.964	watercourse_09448361n	0.947
commercial_bank_08418420n	0.918	flow_02066939v	0.916
central_bank_08349916n	0.914	baltic_coast_09213254n	0.884
loan_13398953n	0.892	tidal_02815241a	0.873

Table 3: Nearest Neighbors Comparison.

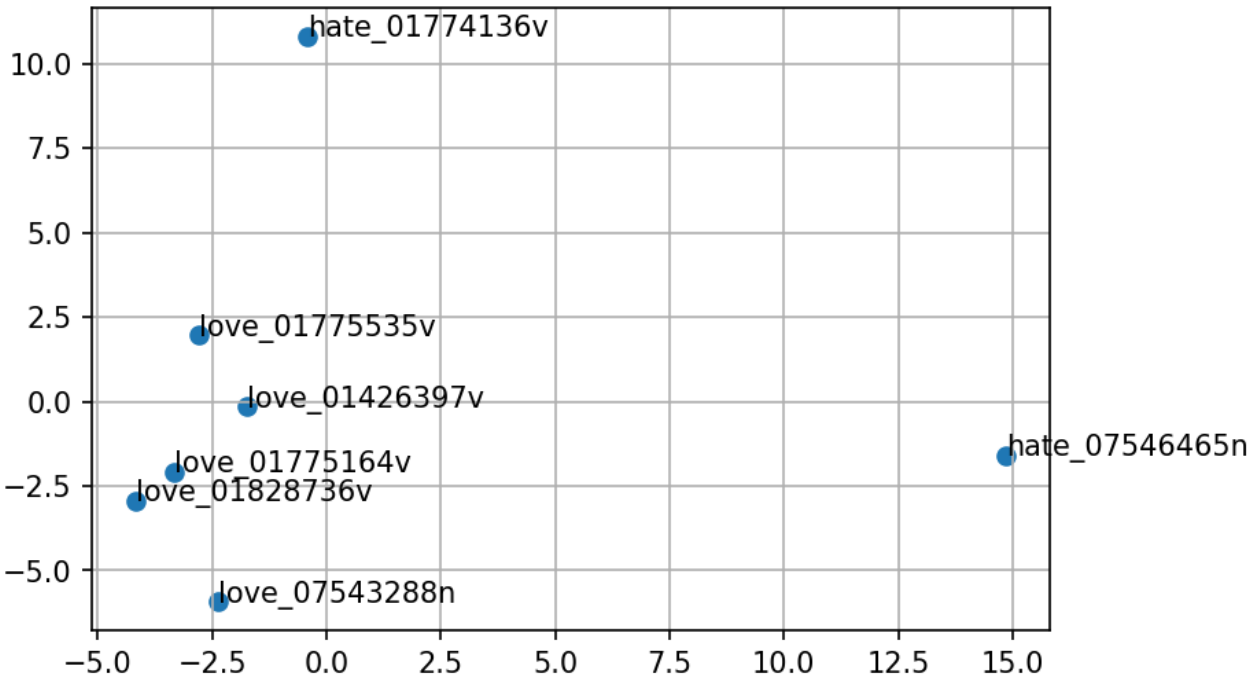

Figure 1: PCA on different meanings of love and hate.

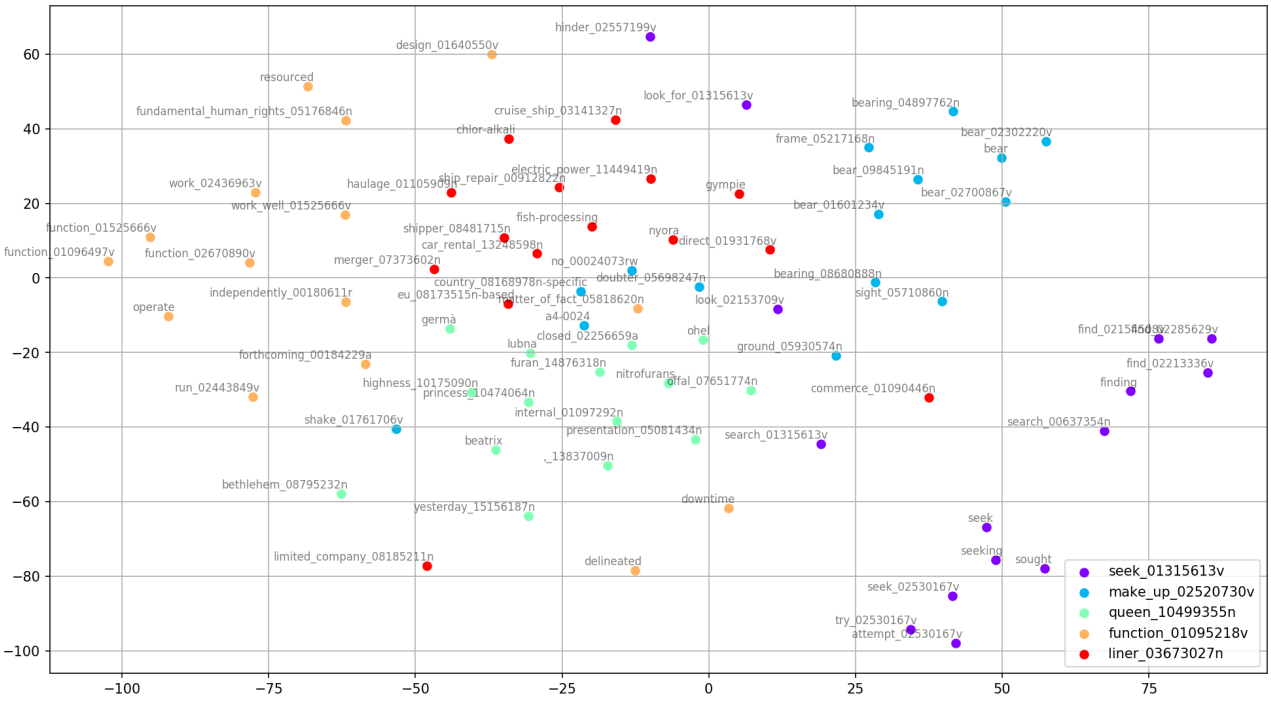

Figure 2: t-SNE plot of word clusters for randomly chosen words from the vocab.