Semantic Correction of Word Embeddings

Motivation

Word embeddings are an essential component of NLP systems. However, almost all static embedding models fail to incorporate semantic information. While there exists contextualized embedding models like BERT, these are prohibitively expensive to train and time consuming. Our project implements a post-processing method that injects semantic information into out-of-the-box embeddings.
We incorporate information from synonyms, antonyms, hypernyms, hyponyms, and meronyms to learn representations that are more semantically accurate while trying to preserve the syntactic integrity of the original embeddings.

Contribution

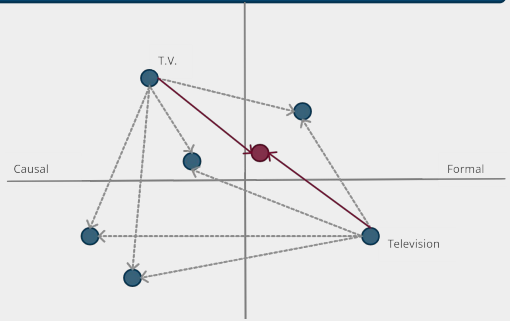
1. We propose semantic correction, an approach that incorporates **prior** information about word-relations as a post-processing step.
2. The approach is applicable to any trained word-representation, as it uses standard information from word banks and little training time.
3. Unlike other post-processing approaches (retro and counter fitting), we account for syntactic relations between all word-pairs in the vocabulary, ensuring minimum loss of corpus-specific syntactic information.

Data

We use 2 pre-trained embeddings that take different approaches to syntax modeling:
1. **GloVe:** Context Window (100k *truncated*)
2. **SynGCN:** Dependency Graph *(150k)*

Syntactic Space Preservation and Semantic Correction

Syntactic representations are generally calculated in isolation. Semantic correction jointly optimizes over both, syntactic loss and semantic loss.
Goal: Move words closer to their semantically relatives, while trying to keep relations between the word and all other words constant - *Vector Space Preservation*.
Syntactic Loss: The pairwise similarity between a word, *i*, and all other words in the corpus. This measures the relative position of a given word in the coordinate space.
This loss is 0 at the beginning of the correction process as it is assumed that the training procedure from the pre-trained embeddings is syntactically optimal.
Semantic Loss: This is a measure of similarity between a word and its semantic relative.
At the beginning of the process, this is >=0 since semantics are not considered in context and dependency-based embeddings.
Optimization: Minimize the weighted sum of semantic and syntactic loss.



$$\mathcal{L}_i^{sem} = \sum_{q_j \in \mathcal{R}} (1 - \cos(q_i, q_j))$$

$$\mathcal{L}_i^{syn} = 1 - \sum_{W} cos(q^*.W^T, q.W^T)$$

$$\mathcal{R} = \{syn, ant, hyper, hypo, mer\}$$

$$min_q \mathcal{L} = \mathcal{L}^{syn} + \gamma.\mathcal{L}^{sem}$$

Standardized Test Results

	Word Similarity			Concept Categorization			Word Analogy	
Method	WS353S	WS353r	SIMLEX999	AP	Battig	BLESS	SemEval12	MSR
Glove	67.1	45.5	38.8	62.1	40.1	**83.5**	15.6	15.6
Glove+	**70.4**	**46.4**	**45.6**	**65.9**	**41.6**	80	**16.4**	**18.2**
SynGCN	75	48.5	44.7	**67.9**	**46.3**	77.5	22.6	52.8
SynGCN+	**76.8**	**49.4**	**48**	67.4	45.6	**79.5**	**23.1**	**54.7**

Better Analogy Test Set (BATS)

BATS	GloVe	GloVe+	SynGCN	SynGCN+
Total	7.6	**7.6**	5.6	**7.2**
[name - occupation]	13	**16.7**	4	**14.3**
[hypernyms - misc]	82.9	**95.7**	76.5	**95.7**
[things - color]	4.1	**10.41**	4.1	**8.3**
[country - language]	1.2	**2.4**	**0.6**	**0.6**

Conclusion

Our task was to generate word representations in an unsupervised setting. Traditional approaches take a text corpus and then learn a k-dimensional embedding for each word in the vocabulary. Overall, we noticed our results outperform SynGCN and all existing word embedding methods. This is perhaps due to other models separating Semantics and Syntactic training whereas we optimizer over both.

Acknowledgement

Github Repo: Source Code
Basis and Inspiration: We derived our hypothesis and methods from the following papers: SynGCN and Retrofitting Word Vectors.

Carrie Hay Kellar, Lukas Geiger, Tanmay Shah