Word Sense Induction and Disambiguation using Context Embeddings

Easwaran Ramamurthy
eramamur@andrew.cmu.edu

Devendra Singh Sachan
dsachan@andrew.cmu.edu

Tejus Siddagangaiah
tsiddaga@andrew.cmu.edu

1 Introduction

Polysemous words are those words which have different but related senses when used in different contexts. In contrast, homonymous words are those words which have totally different senses when used in different contexts. An example of a homonymous word is "bark" which has two completely different senses when used in the statements "My dog would always bark at mailmen" and "The tree's bark was a rusty brown". An example of polysemous word is the occurrence of the word "newspaper" in the two sentences "The newspaper got wet in the rain" and "The newspaper fired some of its editing staff". Disambiguation of word senses is an important task that has several applications including machine translation, information retrieval and question answering. Word sense induction (WSI) aims at identifying the correct number of senses of a word. Word sense disambiguation (WSD) on the other hand aims at annotating the correct induced senses using a standardized glossary such as WordNet or learned sense inventory.

Distributed representations of words using vector space models (VSMs) of lexical semantics have recently gained huge popularity in the field of natural language processing and natural language understanding. VSMs represent words as dense, real-valued vectors in an embedding space where semantically and syntactically similar words are closer to each other. Such representations have been shown to improve generalization on a variety of natural language processing tasks (Bengio et al. [1]; Mnih and Hinton [2]; Collobert and Weston [3]) specially when there is an opportunity to train on very large corpuses. The recent considerable interest in the CBOW and Skip-gram models of Mikolov et al [4], collectively know as word2vec stems from the fact that they are simple log-linear models that can produce high quality word embeddings using an efficient and scalable approach. One notable deficiency in these works is that they cannot capture multiple senses of each word, thereby ignoring polysemes and homonyms. In the next section, we discuss some of the more recent work in this area that focuses on approaching the multi-sense embedding task, followed by a discussion of some very recent approaches that we have experimented upon.

2 Background And Related Work

Firth's distributional hypothesis states that the semantics of a word is reflected in the contexts in which it occurs. Computing multi-sense word embeddings can therefore be useful for tasks such as WSI and WSD. This is exactly the focus of the seminal paper by Reisinger and Mooney [5] where they use efficient unsupervised learning to cluster context-specific word vectors to produce multiple "sense-specific" prototypes for each word which can capture homonymy as well as polysemy. An illustration of a multi-prototype model is shown in Figure 1.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

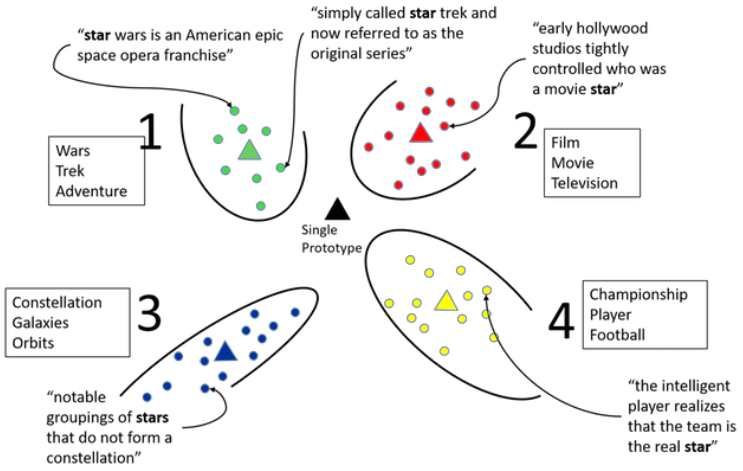

Figure 1: Illustration showing difference between single prototype models and multi-prototype models that capture different sen1ses for different contexts of words.

These multi-prototype models are defined by clustering feature vectors $v(c)$ for each context $c \in C(w)$ that appears in the corpus. These feature vectors can either be *tf-idf* or χ^2 weighted features of words appearing in a window of length W around the given word. Semantic similarity between two isolated words or also two words in context can then be measured using different distance measures like average similarity between all pairs of clusters for two words or maximum similarity calculated between the most likely clusters identified for a word in a given context. However, this approach provided no method to select potentially many vectors that capture context specificity for downstream NLP classification tasks.

Trask et al [6] proposes the "sense2vec" algorithm in which they uses parts of speech (POS) tags to identify the potential number of senses for an ambiguous word. For instance, the word "apple" can have two sense tags "NN" and "NNPS" which can denote the "fruit sense" and "company sense" of the word respectively. This eliminated the need for learning embeddings multiple times and also the clustering step was not required. One drawback with this approach is that it fixes the number of senses using POS tags which may be higher in number as compared to the ground truth. Below we discuss two approaches that try to circumvent this problem.

3 Instance Context Embeddings

Recently, Kågerback et al [7] proposed instance-context embeddings (ICE) that leverage pre-trained word embedding vectors to create context embeddings. This approach demonstrates a novel way to weight context words based on semantic and temporal components. Specifically, the context embedding $(\mathbf{c})$ for a target word instance $\mathbf{i}$ is defined as a weighted function of it's context-word embeddings $\mathbf{v}_c$ as follows:

$$\mathbf{c}_i = \frac{1}{Z} \sum_{-T \leq c \leq T; c \neq 0} \psi_{i,c} \mathbf{v}_c \tag{1}$$

Here T is the context width and the weighting $\psi_{i,c}$ is defined as the product of a semantic and temporal component as follows:

$$\psi_{i,c}^{ice} = \psi_{i,c}^{semantic} \psi_{i,c}^{temporal} \tag{2}$$

where the semantic component is defined as:

$$\psi_{i,c}^{semantic} = \frac{1}{1 + e^{-\mathbf{v}_c \mathbf{u}_i}} \tag{3}$$

Here, $\mathbf{u}_i$ is the outer word embedding representing target word. The temporal component weights are defined using distance of context words to the target word as follows:

$$\psi_{i,c}^{temporal} = \frac{1}{T}\max(0, T - |i - c|) \tag{4}$$

This kind of weighting captures both semantic relationships and distance of context words from target word in the corpus.

4 Linear Combinations of Different Senses

Another recent work is that of Arora et al [8], where they conducted an interesting experiment in which they replace every occurrence of any two random words in the corpus (w_1 and w_2) with a single word (w_{new}). Following this, the authors compute an embedding for words in the corpus. The embeddings on the original corpus are then compared to the embeddings obtained from this modified corpus. Authors then compare the embedding vector of the new word $v_{w_{new}}$ with linear combinations of v_{w1} and v_{w2}. They selected w_1 and w_2 with varying frequency ratio and repeated the experiment multiple times. Authors observed that $v_{w_{new}}$ always lied in the subspace spanned by v_{w_1} and v_{w_2}. Authors conclude that $v_{new} \approx \alpha v_{w_1} + \beta v_{w_2}$. This allows us to express each word as a linear combination of all its senses as follows:

$$v_{\text{apple}} \approx \alpha_1.v_{\text{apple}_1} + \alpha_2.v_{\text{apple}_2} + \alpha_3.v_{\text{apple}_3} + \cdots \tag{5}$$

Here, apple_i represents the i-th sense of the word apple in the corpus. In the RAND-WALK work Arora et al. [9] propose a random walk on atomic discourses approach according to which directions in the embedding space correspond to local topic structure (discourse) in the corpus . The authors state that the text corpus is being generated by a random walk process on discourse vectors. At discourse c_t, their model outputs words using a log-linear function $Pr[w \mid c_t] \propto \exp(c.v_w)$. The sense vectors for a polysemous word can be recovered from this model by solving a sparse coding optimization problem [10] as follows:

$$\begin{aligned} \min_{\alpha_i^{(j)}, \phi_i} \sum_{j=1}^{m} \left\| \mathbf{v_w}^{(j)} - \sum_{i=1}^{k} \alpha_i^{(j)} \phi_i \right\|^2 + \lambda \sum_{i=1}^{k} S(\alpha_i^{(j)}) \\ \text{subject to} \qquad \|\phi_i\|^2 \leq C, \quad for \quad i = 1, 2, ...k \end{aligned} \tag{6}$$

Here $\mathbf{v_w^{(j)}}$ represents the word embedding for the j-th word in the vocabulary. The loss term above evaluates how much the reconstruction from sparse coding deviates from the true embedding. λ is a scaling constant to determine the relative importance of these two contributions. $S(.)$ is a sparsity cost function which penalizes α_i and induces it close to zero, thereby forcing the representation to be sparse. Least angle regression (LARS) [11] is used to learn the α_i's and ϕ_i's allowing at most l of the α_i's to be non zero for a particular vector. After learning, the ϕ_i's with $\alpha_i \neq 0$ are analyzed to determine the word's sense.

5 Experiments

We experimented with both the approaches described above and discuss the results obtained. We present a qualitative analysis of the nearest neighbors for both approaches as discussed below.

5.1 Instance-Context Embeddings (ICE)

We chose Wikipedia corpus (June 2015 dump) for initial experiments. We computed Inverse Document Frequency (IDF) values for every word in the corpus. For every article in Wikipedia, we split it into its corresponding sentences using the splita [12] library. Then, we filter every sentence in which ambiguous words occur. We follow the ICE approach in order to learn the context embeddings. In

addition to semantic and temporal weights, we found that including IDF values as a weight feature results in improved similarity scores. We selected the window size as 3 context words and normalized the context embeddings to have a norm of 1. We use spherical K-Means algorithm to cluster the context vectors. We use the resulting centroids as sense vectors for the ambiguous word.

5.2 Sparse Coding

We used pretrained word vectors learned from Wikipedia (2014 dump) and Gigaword5 corpus available by applying Global Vectors [13] [1]. We select the value of C to be 1 and the maximum number of non-zero coefficients l to be 5. We found that the LARS dictionary learning algorithm was quite slow for all the word vectors. Therefore, we restricted the vocabulary size to the most frequent 10000 words using $k = 500$.

6 Evaluation of Method

We do a qualitative evaluation and display the nearest neighbour of every word sense. We observe that we get semantically similar words for each sense by using the approach of context embeddings. In sparse coding approach, we see that word senses are recovered if we keep similarity threshold as 0.5 and ignore those ϕ_i's where similarity is below this threshold.

From Table1 and Table2, we observe that, sparse coding recovers two main senses of the word "apple" and "star" after applying a similarity threshold. Also, for the ICE method the results are summarised in Table3 and Table4 and we observe that, the algorithm discovers main senses of the ambiguous words after context clustering.

Apple$_1$		**Apple**$_2$		**Apple**$_3$		**Apple**$_4$		**Apple**$_5$	
afl	0.178	warren	0.174	ceo	0.673	trees	0.742	deportation	0.153
ashcroft	0.176	patricia	0.170	ibm	0.664	tree	0.709	1924	0.152
brisbane	0.167	senegal	0.163	intel	0.660	flowers	0.705	1921	0.150
herbert	0.152	harrison	0.162	microsoft	0.646	fruit	0.659	ricardo	0.145
asserted	0.140	jenkins	0.161	citigroup	0.642	flower	0.641	yah	0.142
aviation	0.138	shane	0.155	executives	0.598	leaf	0.635	kurdistan	0.139
surrendered	0.136	davies	0.141	yahoo	0.586	stems	0.630	deported	0.137
mclaren	0.136	michelle	0.141	cola	0.585	fruits	0.629	1922	0.137
assisted	0.135	trinidad	0.141	corp.	0.581	leaves	0.602	1926	0.136
dissident	0.134	daughters	0.140	aol	0.571	planted	0.586	1891	0.136

Table 1: Word senses for "apple" identified using sparse coding

Star$_1$		**Star**$_2$		**Star**$_3$		**Star**$_4$		**Star**$_5$	
guest	0.714	20003	0.255	temples	0.185	particles	0.612	digit	0.164
comedian	0.667	s.e.	0.186	overview	0.163	earth	0.610	mercantile	0.156
host	0.635	disambiguation	0.177	decrease	0.159	planet	0.608	municipalities	0.140
celebrity	0.621	datafile	0.161	libraries	0.159	solar	0.585	consolidated	0.136
appearing	0.616	sah	0.146	regulated	0.157	magnetic	0.560	populous	0.131
featured	0.614	——	0.142	permits	0.157	observations	0.555	territorial	0.127
hosted	0.607	hah	0.140	layers	0.152	radiation	0.549	yushchenko	0.124
show	0.600	kah	0.132	steep	0.150	optical	0.544	pence	0.122
celebrities	0.585	bah	0.111	workshops	0.149	object	0.542	yemeni	0.120
starred	0.567	—-	0.107	reactor	0.146	gravity	0.529	emirates	0.118

Table 2: Word senses for "star" identified using sparse coding

[1] http://nlp.stanford.edu/projects/glove/

Apple$_1$		**Apple**$_2$	
fruit	0.398	apple	0.478
cherry	0.343	macintosh	0.467
apples	0.337	iphone	0.447
apple	0.334	software	0.442
mango	0.333	ipad	0.423
trees	0.327	pc	0.421
honey	0.327	desktop	0.42
juice	0.324	ipod	0.416
fruits	0.324	computers	0.41
peach	0.323	microsoft	0.409

Table 3: Word senses of "apple" identified using ICE

Plants$_1$		**Plants**$_2$	
power	0.664	species	0.69
electricity	0.522	genus	0.633
nuclear	0.486	flowering	0.618
generating	0.477	endemic	0.531
hydroelectric	0.46	genera	0.52
energy	0.456	asteraceae	0.476
reactors	0.43	plants	0.47
fuel	0.43	shrubs	0.462
plant	0.422	shrub	0.459
capacity	0.418	subspecies	0.457

Table 4: Word senses of "plants" identified using ICE

7 Work division and Timeline

In the project proposal, we mentioned that our plan was to replicate the work of Yuan et al [14], Neelkantan et al [15] and Trask et al [6]. However upon further research, we find that the works we mentioned above seem to be more promising and therefore we experimented with them. We observed that the approach of Trask et al has been replicated in SpaCy tool [2] and a nice interface is available online [3] to view the results trained on Reddit's dataset collected over 1 year. As an online interface was available, we do not mention its results in this report.

We will do quantitative analysis of the above algorithms on SCWS [4](by computing Spearman's correlation coefficient), SemEval 2013, Task 13 [5] and MSH dataset [6]. Detailed descriptions of these datasets are provided in the project proposal. The plan is to have baselines on each of these datasets for both of the above mentioned algorithms. We will then work on an algorithm to learn context embeddings using CNN and LSTM as opposed to hand engineering using linear combinations and will then evaluate our algorithm on the above datasets. We will split the work related to the implementation of the above tasks among the team members.

[2]https://spacy.io/

[3]https://demos.explosion.ai/sense2vec/

[4]http://ai.stanford.edu/ ehhuang/

[5]https://www.cs.york.ac.uk/semeval-2013/task13.html

[6]https://wsd.nlm.nih.gov/collaboration.shtml

References

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *journal of machine learning research*, 3(Feb):1137–1155, 2003.

[2] Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM, 2007.

[3] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

[4] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[5] Joseph Reisinger and Raymond J Mooney. Multi-prototype vector-space models of word meaning. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 109–117. Association for Computational Linguistics, 2010.

[6] Andrew Trask, Phil Michalak, and John Liu. sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings. *arXiv preprint arXiv:1511.06388*, 2015.

[7] Mikael Kågebäck, Fredrik Johansson, Richard Johansson, and Devdatt Dubhashi. Neural context embeddings for automatic discovery of word senses. In *Proceedings of NAACL-HLT*, pages 25–32, 2015.

[8] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *arXiv preprint arXiv:1601.03764*, 2016.

[9] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Rand-walk: A latent variable model approach to word embeddings. *arXiv preprint arXiv:1502.03520*, 2015.

[10] Bruno A Olshausen et al. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[11] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online dictionary learning for sparse coding. In *Proceedings of the 26th annual international conference on machine learning*, pages 689–696. ACM, 2009.

[12] Dan Gillick. Sentence boundary detection and the problem with the us. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 241–244. Association for Computational Linguistics, 2009.

[13] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[14] Dayu Yuan, Ryan Doherty, Julian Richardson, Colin Evans, and Eric Altendorf. Word sense disambiguation with neural language models. *arXiv preprint arXiv:1603.07012*, 2016.

[15] Arvind Neelakantan, Jeevan Shankar, Alexandre Passos, and Andrew McCallum. Efficient non-parametric estimation of multiple embeddings per word in vector space. *arXiv preprint arXiv:1504.06654*, 2015.

[16] Huazheng Wang, Fei Tian, Bin Gao, Jiang Bian, and Tie-Yan Liu. Solving verbal comprehension questions in iq test by knowledge-powered word embedding. *arXiv preprint arXiv:1505.07909*, 2015.

[17] Fei Tian, Hanjun Dai, Jiang Bian, Bin Gao, Rui Zhang, Enhong Chen, and Tie-Yan Liu. A probabilistic model for learning multi-prototype word embeddings. In *COLING*, pages 151–160, 2014.

[18] Jiwei Li and Dan Jurafsky. Do multi-sense embeddings improve natural language understanding? *arXiv preprint arXiv:1506.01070*, 2015.