Literature Review

Word embedding is a technique in Natural Language Processing (NLP) that transforms the words in a vocabulary into dense vectors of real numbers in a continuous embedding space. One topic involving the application of word embedding that has obtained much research interest is diachronic semantic change. For example, Yao et al. (2018) developed a dynamic statistical model that learns time-aware word vector representation to investigate the changing meanings and associations of words throughout time. They proposed a model which simultaneously learns word embeddings and aligns them across time. Given $D = (D_1, ..., D_T)$ where each D_t is the corpus of all documents in the t-th time slice, and an overall vocabulary $V = \{w_1, \ldots, w_V\}$ of size V, they computed the $V \times V$ pointwise mutual information (PMI) matrix specific to a corpus D, whose w, c-th entry is: $PMI(D, L)_{w,c} = \log((\#(w, c)\cdot|D|)/\#(w)\cdot\#(c))$, where #(w, c) counts the number of times that words w and c co-occur within a window of size L in corpus D, and #(w), #(c) counts the number of occurrences of words w and c in D; |D| is total number of word tokens in the corpus; L is typically around 5 to 10. Then for each time slice t, they defined the w, c-th entry of positive PMI matrix (PPMI(t ,L)) as: $PPMI(t, L)_{w,c} = \max\{PMI(D_t, L)_{w,c}, 0\}$.: = Y(t). The temporal word embeddings U (t) must satisfy $U(t)U(t)^T \approx$ PPMI(t, L). Thus, they claimed finding temporal word embeddings as the solution of the following joint optimization problem:

$$\min_{U(1),\ldots,U(T)} \frac{1}{2}\sum_{t=1}^{T} \|Y(t) - U(t)U(t)^T\|_F^2 + \frac{\lambda}{2}\sum_{t=1}^{T}\|U(t)\|_F^2 + \frac{\tau}{2}\sum_{t=2}^{T}\|U(t-1) - U(t)\|_F^2,$$

where Y(t) = PPMI(t, L) and $\lambda, \tau > 0$. The optimization problem was solved by using a scalable block coordinate descent method. They applied the model to a dataset based on 99,872 articles from the New York Times, published between January 1990 and July 2016, which were divided into T = 27 partitions based on yearly time slices and consisted of V = 20, 936 unique words. They performed a grid search to find the best regularization and optimization parameters, and obtained = 10, $\tau = \gamma = 50$, and ran for 5 epochs. All distances between two words were calculated by the cosine similarity between embedding vectors. For evaluation, their designed qualitative (trajectory visualization, equivalence searching, and popularity determination) and quantitative methods (semantic similarity, alignment quality, and robustness) showed that their dynamic embedding method performed favorably against other temporal embedding approaches.

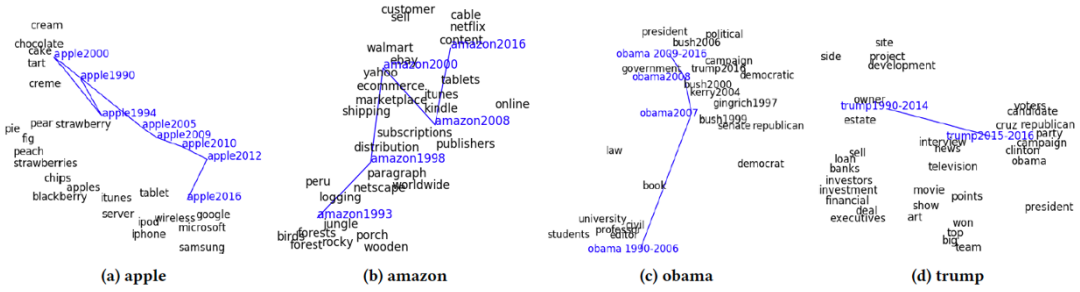

Figure 1. Trajectories of brand names and people through time from Yao et al.'s (2018) paper

Garg et al. (2018) also used word embeddings to investigate semantic change over time, and they showed the technique could be combined with a social sciences topic: they developed a framework to demonstrate how the temporal dynamics of the embedding quantifies changes in gender and ethnic stereotypes in the last 100 years. They used the standard Google News word2vec vectors trained on the Google News dataset for contemporary snapshot analysis, and used previously trained Google Books/Corpus of Historical American English (COHA) embeddings for historical temporal analysis. They also used the GLoVe algorithm to train embeddings from the New York Times Annotated Corpus for every year between 1988 and 2005 as additional validation. Then they collated several word lists to represent each gender (men, women) and ethnicity (White, Asian, and Hispanic), as well as neutral words (adjectives and occupations). For occupations, historical US census data were used to extract the percentage of workers in each occupation that belong to each gender or ethnic group, which was then compared to the bias in the embeddings. Thus, they were able to measure the strength of association (embedding bias) between neutral words and a group using the embeddings and word lists: they computed the representative group vector by taking the average of the vectors for each word in the given gender/ethnicity group, and then computed the average Euclidean distance between each representative group vector and each vector in the neutral word list of interest (occupations or adjectives). The difference of the average distances would be the metric for bias. In order to verify that the bias in the embedding accurately reflects sociological trends, they compared the trends in the embeddings with quantifiable demographic trends in the occupation participation and historical surveys of stereotypes. Results showed that changes in the embedding tracks closely with demographic and occupation shifts over time, and also illuminates how specific adjectives and occupations became more closely associated with certain populations over time.

Compared to semantic change over time, the topic of semantic change over domains has been less studied, despite much research on word embeddings in specific domains. Lee et al. (2019) investigated how the pre-trained language model BERT can be adapted for biomedical corpora. They introduced BioBERT (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining), a domain-specific language representation model pre-trained on large-scale biomedical corpora. First, they initialized BioBERT with weights from BERT, which was pre-trained on general domain corpora (English Wikipedia and BooksCorpus). Then, BioBERT was pre-trained on biomedical domain corpora (PubMed abstracts and PMC full-text articles). To show the effectiveness of the approach, BioBERT was fine-tuned and evaluated on three popular biomedical text mining tasks: biomedical named entity recognition, biomedical relation extraction, and biomedical question answering, where BioBERT largely outperforms BERT and previous state-of-the-art models. Zhang et al. (2019) presented an open set (BioWordVec) of biomedical word vectors/embeddings that combines subword information from unlabeled biomedical text with a widely-used biomedical controlled vocabulary called Medical Subject Headings (MeSH). They pointed out two limits of traditional biomedical word embedding: 1) most of them were trained using the word2vec1 or GloVe model7, which use a distinct vector to represent each word, and tend to ignore the internal structure of words and are not good at learning rare or out of vocabulary (OOV) words in the training data. 2) existing word embeddings mainly focus on using the single source of large text corpora in PubMed. So they created a new set of word embeddings using a subword embedding model on two different data sources. To build this model, they first constructed a MeSH term graph using SPARQL queries, in which each word is a node and connected by

undirected edges if they have relations. Then, they transferred the relations of the MeSH term graph into ordered sequences of the heading nodes by using a random walk procedure called node2vec. As for the subword embedding model, They modified the continuous skip-gram model to learn the character n-grams distributed embeddings. Character n-grams means that each word is represented as the sum of the vector representations of its n-grams, and it will improve the embedding quality since there is a large amount of compound words in the biomedical domain. The objective function was based on the function of skip-gram model, and was listed below:

$$J = J_{PubMed} + J_{MeSH}$$

In which

$$J_{PubMed} = \frac{1}{T}\sum_1^T \sum_{c \in C_t} \log p(w_c|w_t)$$

$$J_{MeSH} = \frac{1}{N}\sum_1^N \sum_{c \in C_t} \log p(D_c|D_t)$$

Where w_i is the sequence of words; D_i is the sequence of main-heading nodes; C_t is the set of the surrounding words of w_t; N is the total number of MeSH main headings; T is the total vocabulary size. P is defined as the probability of observing its surrounding word w_c:

$$p(w_c|w_t) = \frac{e^{s(w_t, w_c)}}{\sum_{j=1}^{W} e^{s(w_t, w_j)}}$$

The scoring function was defined as a scalar product, but the two vectors may no longer be distinct, so the new scoring function is defined as $\sum_{g \in (1,\ldots,G)} z_g^T v_c$, where (1, …, G) is the n-gram set of w_t, z_g is the vector of character n-gram g, and v_c is the vector of word w_c. The result shows that the cosine similarity score calculated by their word embedding is higher than the other word embeddings they referenced, and their method capture the similar words of compound words better than the methods they referenced, which indicates their method can make good use of the sub-word information and internal structure of words. Dollah et al. (2019) used Convolutional Neural Network to perform classification of biomedical text. First, they performed feature extraction on 11,566 biomedical abstracts from the Ohsumed dataset, and a list of unique features was generated. All the features were then added to the multiword tokenizer lexicon for tokenizing phrases or compound words. A multiword tokenizer instead of a single word tokenizer was used because of a special property of biomedical abstracts: almost all the biomedical terms were compound terms. Next, a word embedding layer was set up with Keras in order to transform all the words in the text with the same or similar meaning to have a similar representation in the form of a vector. The sequence of embedding vectors obtained was then converted into a compressed representation, which became the input for the stack of convolutional layer and max-pooling layer. They used ReLU as the activation function, and softmax function with categorical_crossentropy loss function to solve the multi-class classification problem. During the training process, the cross-validation method was used to reduce problems such as overfitting and give an insight on how well the model can generalize to each independent data set. Parameters used in the model included epoch = 10, filter size = 128, kernel size = 3, and batch size = 128. A result of 54.79% average accuracy, 61.00% average precision, 60.00% average recall and 60.50% average F1-score was obtained.

Studies of word meaning across domains have been mainly concerned with domain specific word sense disambiguation or term ambiguity detection. For instance, Maynard et al. (1998) developed a method for automatic term sense disambiguation based on the identification of relevant contextual information from sublanguage corpora. They

combined basic semantic roles derived from the corpus with domain-specific semantic categories from UMLS, a specialized medical thesaurus, and used an EBMT-based matching algorithm to compare related terms. For example, the term *actinic keratosis* would be tagged with *acquired abnormality*, its semantic type from the UMLS Metathesaurus. Similarity of calculated as following:

$$\mathrm{sim}(w_1 ... w_n) = \frac{n(\mathrm{com}(w_1 ... w_n))}{\mathrm{pos}(w_1 ... w_n)}$$

Where n is the number if words being compared; com($w_1 \ldots w_n$) is the commonality weight of words 1…n; pos($w_1 \ldots w_n$) is the positional weight of words 1…n. Wang et al. (2013) studied the problem of lexical ambiguity detection and proposed methods that can automatically identify potentially ambiguous concepts in software requirement specifications. Specifically, they focused on two types of lexical ambiguities, i.e., Overloaded and Synonymous ambiguity. For overload ambiguities, they studied the following features that measure the diversity of the context for a concept: concept frequency, context diversity, number of clusters in the context, and inter-cluster distance. For synonymous ambiguities, they combined the following features: context-based similarity, pattern-based similarity, and textual based similarity. Experiment results over four real-world software requirement collections showed that the proposed methods are effective in detecting ambiguous terminology.

Ferrari et al. (2017) is almost the only notable work so far for explicitly measuring across domain meaning shifts. They aimed to estimate the degree of ambiguity of typical computer science nouns when used in different application domains. First, they crawled Wikipedia to extract CS documents and domain specific documents for a given domain, and pre-processed the documents to ease the subsequent steps. Then, they searched for the most frequent nouns in the CS documents. These nouns were those whose meanings they hoped to compare. Each occurrence of these nouns in the domain specific documents was replaced with a uniquely identifiable modified version of the noun (the noun is prefixed by an underscore character). They applied the word2vec algorithm on the corpus composed by the CS documents and the domain specific documents to learn the word embeddings. Finally, they measured the similarity between the embeddings of the frequent nouns in CS and the embeddings of their modified versions. Their preliminary experiments, performed on five different domains, showed promising results.

Another topic of importance for word embeddings is their evaluation. Wang et al. (2019) conducted an extensive evaluation on a large number of word embedding models for language processing applications [2]. First, they introduced popular word models, which are Neural Network Language Model (NNLM), Continuous-Bag-of-Words (CBOW) and Skip-Gram, Co-occurrence Matrix, FastText, N-gram Model, Dictionary Model and Deep Contextualized Model. Then they discussed some desired properties of good embedding models (non-conflation, robustness against lexical ambiguity, demonstration of multifacetedness, etc.) and of evaluators (good testing data, comprehensiveness, high correlation, etc.). Then they divided evaluators into two types: intrinsic and extrinsic evaluators, and discussed several evaluators of both types, followed by the experimental results. For intrinsic evaluators, five evaluators were discussed, which are 1) word similarity: they mainly introduced cosine similarity, which is defined by $cos(\omega x, \omega y) = \omega x \times \omega y / \|\omega x\| \|\omega y\|$, where ωx and ωy are two word vectors and $\|\omega x\|$ and $\|\omega y\|$ are the $l2$ norm; 2) word analogy: they discussed 3CosAdd and 3CosMul method; 3) concept categorization: the ability to split a given set of words into different categorical subsets, which can be tested by implementing a clustering

algorithm on word vectors; 4) outlier detection: tests the semantic coherence of vector space models by taking a set of words $W=\omega 1,\omega 2,...,\omega n+1$where there is one outlier and take a compactness score of word ωas $c(\omega)=1n(n-1)\Sigma\omega i\in W\backslash\omega\Sigma\omega j\in W\backslash\omega sim(\omega i,\ \omega j)$ in which sim is the pairwise semantic similarity, then the outlier is the word with the lowest compactness score; and 5) QVEC: an intrinsic evaluator that measures the component-wise correlation between word vectors from a word embedding model and manually constructed linguistic word vectors in the SemCor dataset. They selected six word embedding models: SGNS, CBOW, GloVe, FastText, ngram2vec and Dict2vec, and trained them on the same corpus – wiki2010 (6G, without XML tags). The threshold for vocabulary is set to 10. The results : 1). Word similarity: they chose 13 datasets with different number of word pairs and the ngram2vec has the best performance generally with a top word similarity for 7 in 13 test datasets; 2). Word analogy: SGNS has the best performance for both datasets which contain a large number of analogy questions. 3). concept categorization: SGNS-based evaluators (including SGNS, ngram2vec and dict2vec) perform better than others. 4). Outlier deduction: the result is not consistent as the evaluators have a big difference on the chosen database, and it was discussed later. 5) QVEC: they used QVEC toolkits and ngram2vec have best overall results. For extrinsic evaluators, five evaluators were discussed, which are 1) part-of-speech tagging, 2) chunking, 3) named-entity recognition, 4) sentiment analysis and 5) neural machine translation. For the above 5 evaluators, SGNS-based models tend to work better than other models, but for most evaluators, the performance differences are small and may be affected by the properties of datasets or other factors.

In the present study, we measure how word sense varies with the domain in which the word occurs. We use the methods previously employed by Yao et al. (2018) to measure differences in word sense over time, since their model has proven to be successful, outperforming other models by both qualitative and quantitative evaluations. We also adopt evaluations similar to those in Garg et al. (2018), which provided a effective way to test how well the shift in senses in word embeddings correlates with what's happening in the real world.

Works Cited

Rozilawati Dollah, Chew Yi Sheng, Norhawaniah Zakaria, Mohd Shahizan, and Abd Wahid Rasib. 2019. Deep learning classification of biomedical text using Convolutional Neural Network. *International Journal of Advanced Computer Science and Application* 8, 10 (2019), 512-517.

Alessio Ferrari, Beatrice Donati, and Stefania Gnesi. 2017. Detecting domain-specific ambiguities: An NLP approach based on wikipedia crawling and word embeddings. In *Proceedings of the 2017 IEEE 25th International Requirements Engineering Conference Workshops*, pages 393–399.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2017. Word embeddings quantify 100 years of gender and ethnic stereotypes. *arXiv:1711.08412* (2017).

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *arXiv:1901.08746* (2019).

Diana Maynard and Sophia Ananiadou. 1998. Term sense disambiguation using a domain-specific thesaurus. In *Proceedings of 1st International Conference on Language Resources and Evaluation*, pages 681–687, Granada, Spain.

Bin Wang, Angela Wang, Fenxiao Chen, Yuncheng Wang, C.-C. and Jay Kuo. 2019.

Evaluating word embedding models: methods and experimental results. *arXiv:1901.09785* (2019).

Yue Wang, Irene L. Manotas Guti´errez, Kristina Winbladh, and Hui Fang. 2013. Automatic detection of ambiguous terminology for software requirements. In *Proceedings of the International Conference on Application of Natural Language to Information Systems*, pages 25–37.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. *arXiv:1703.00607* (2018).

Yijia Zhang, Qingyu Chen, Zhihao Yang, Hongfei Lin, and Zhiyong Lu. 2019. BioWordVec, improving biomedical word embeddings with subword information and MeSH. *Scientific Data* 6, 52 (2019). https://doi.org/10.1038/s41597-019-0055-0