3.5.2 DWE Evaluation

In this section, we evaluate effectiveness of our proposed diachronic word embedding modela public test set [178]. The goal is to examine if our proposed DWE model can capture the changing sense of words over time. In all cases, we keep the dimension of word representations as 200.

3.5.2.1 Evaluation Data

We retrieved a newspaper dataset of New York Times from the previous publication [178]. The dataset contains 99,872 newspaper articles in 27 years ranging from 1990 to 2016. We used *RegexpTokenizer* from NLTK [12] to tokenize the documents that only contain alphabets and numbers. We then dropped any paragraphs that are less than 5 tokens and filtered out any words if their frequency are smaller than 5 in the general domain. With the time information, we grouped documents in every three years: 1990-1992 (90-92); 1993-1995 (93-95); 1996-1998 (96-98); 1999-2001 (99-01); 2002-2004 (02-04); 2005-2007 (05-07); 2008-2010 (08-10); 2011-2013 (11-13); 2014-16 (14-16). We show the details of preprocessed evaluation data in Table 3.4. We can observe that the frequency of unique word numbers vary across time domains. Such variations can impact on the vocabulary size and contexts of words, which are important to train word embeddings.

time interval	90-92	93-95	96-98	99-01	02-04	05-07	08-10	11-13	14-16	ALL
#doc	9770	9741	9975	12209	12241	11360	12240	12376	9938	99850
#uw	41688	42213	43017	48011	52016	51000	50522	53368	50229	128333
#awpd	590	600	600	615	710	692	712	781	849	686

Table 3.4: Data stats across each temporal and the general domain, where *#doc* is the number of documents, *#uw* refers to the unique numbers of words in the domain and *#awpd* means the average number of words per document.

3.5.2.2 Baselines

Static. We train the word2vec model [122] on the entire corpus without considering time information. The training parameters keep their default values in the Gensim [136].

Linear. Kulkarni et al. [106] proposed a linear alignment method. The approach first extracts k nearest neighbors of words in both source and target domains and then uses linear regression model to minimize the distance between those word representations. We use the *NearestNeighbors* from scikit-learn [127] and *WLS* function from StatsModels to implement the baseline.

Procrustes. Hamilton et al. [66] used orthogonal Procrustes to align yearly embedding models. We implement the model via *procurstes* function in SciPy [93]. We align embedding models of other temporal intervals towards the embedding model in the latest time interval.

Hierarchy. Zhang et al. [187] developed a weighted linear transformation method weighted by hierarchical clusters of words. To build transformation matrices, the method first chooses top 5% frequent terms in the intersections of vocabularies among the corpora across temporal domains. Then it calculates the weights by the hop distance in the clusters. Finally, the method can learn weighted transformation matrices to align temporal domains together. To implement the model, we used *Ridge* and *linkage* functions from scikit-learn [127] and SciPy [93] respectively. We set the k as 0, weight of l2-norm as 0.05 and leave the rest parameters as default.

DW2V. Yao et al. [178] designed a joint optimization problem that minimizes distances between the embeddings and positive point-wise mutual information (PPMI) and distances between every pair of temporal domain's static embeddings. We follow the same parameter settings in their source codes.[3]

3.5.2.3 Evaluation Results

We present the evaluation method, metrics and results in this section. The evaluation set is from the *DW2V* [178], which measures if the embedding models can categorize words their correct meanings across yearly domains. The test set contains $1,888$ entries with three different columns: word, section label and year. The section label has 11 different categories and indicates close meanings of words at the certain years in New York Times, for example, the "adobe" is under "Technology" section label in 2011. The year is the time information when the word and section

[3] https://github.com/yifan0sun/DynamicWord2Vec

label were generated. We adjust the year label to the time domain schema. For example, if the year is 2013, we will convert it to 2011-13.

$$F_\beta = \frac{(\beta^2 + 1) * P * R}{\beta^2 * P + R}$$

We use the F_β to measure effectiveness of the embedding models on the test set. By using *SpectralClustering* in scikit-learn [127], we first apply the clustering algorithm to group the test words with three different cluster sizes, 10, 15 and 20. We set the affinity as cosine and leave other parameters as defaults. To measure the cluster quality, we then select every two words from the test sets without repetition. We count the pair as a correct option if the two words share the same section label and from the same cluster or if two words are not the same section label and from the different clusters. Otherwise, we will count the selection as a wrong option. Finally, we can apply the F_β to measure the quality of clusters and compare the embedding models at Table 3.5.

Method	10 Clusters	15 Clusters	20 Clusters
Static	.657	.668	.655
Linear	.702	.719	.765
Procrustes	.628	.681	.668
Hierarchy	.636	.809	.878
DW2V	.809	.843	.854
Ours	**.810**	**.916**	**.905**

Table 3.5: Performance evaluation using F_β.

Our method consistently outperforms the other methods and gain about 3.37 absolute percentage improvements. We can also find that a larger cluster generally has better results than a smaller cluster number.