Exploring the Role of Contextualization in Dynamic Contextualized Word Embeddings

Anubhav Agarwal
Princeton University
anubhava

Rohan Jinturkar
Princeton University
rohanj

Henrique Schechter Vera
Princeton University
hvera

Abstract

Contemporary natural language tasks often use embeddings to extract information from text, especially for downstream applications. However, many embedding techniques are non-dynamic, meaning they do not capture extralinguistic context like time and social space. There are techniques that incorporate linguistic context (e.g., large language models like BERT and GPT), and dynamic embeddings can partially incorporate time and social space. We seek to replicate the findings of Hofmann et al. (2020), who propose Dynamic Contextualized Word Embeddings (DCWEs) as a technique for incorporating both linguistic and extralinguistic context by combining a pretrained contextual language model and a joint model of time and social space. After replicating their findings, we extend their work to a downstream sentiment analysis task in two ways: we consider multiple contextual models (e.g., GPT), and we explore the effect of a dynamic component on non-contextual embeddings created by traditional embeddings systems (e.g., Word2Vec). While we replicate the baseline findings, we find that the addition of DCWE is not beneficial for the other contextual embeddings. Furthermore, non-contextual embeddings performed substantially worse than the baseline.

1 Introduction

Word embeddings are a critical development in expanding the scope of the tasks addressable by NLP. A litany of embedding techniques now exist. Traditional methods like Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) compute embeddings statically, representing the word's use in an entire corpus with a singular vector. While helpful in representing text in certain tasks, this approach to language modeling fails to account for changes in immediate textual context (e.g., the same word having different meanings in different sentences), as well as extralinguistic factors that may affect the text being considered (e.g., temporal or social variations).

Other models have been developed that aim to solve both issues. Contextual models, like BERT (Devlin et al., 2019), GPT (Radford et al., 2019), and ELMo (Peters et al., 2018) create contextual embeddings that vary based on the immediate linguistic context of the text that the word occurs in. Furthermore, techniques known as dynamic embeddings represent words as vectors that change based on time (Rudolph and Blei, 2018) and social context (Zeng et al., 2018). However, the two families of techniques exist independently, posing several challenges. Not only is it difficult to ascertain which type of model is preferable for an individual task (e.g., should a system use dynamic or contextual embeddings), but most models have specific limitations that prevent them from capturing the full scope of any dataset (for example, most dynamic embedding techniques can capture *either* temporal or social context, but not both).

With the growing prominence of embeddings for downstream NLP tasks, it becomes increasingly important to combine the benefits offered by individual techniques to better capture the semantic meaning of a corpus. The work by Hofmann et al. (2020) represents a significant step in pursuit of that goal. In their proposed system of Dynamic Contextualized Word Embeddings (DCWEs), non-contextualized embeddings are first modified to incorporate extralinguistic contexts and then passed through a large Pretrained Language Model (PLM) to incorporate linguistic context. This pipeline enables the creation of embeddings that are both dynamic and contextual in nature. Notably, the authors observe general improvements across the board in masked language modeling perplexity, as well as in sentiment analysis performance.

The performance of the model technique proposed by Hofmann et al. (2020) offers interesting conclusions regarding embedding creation. In this

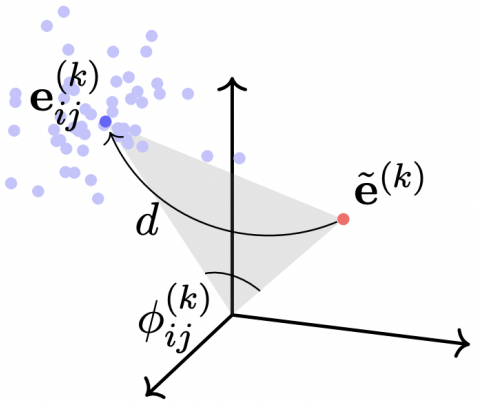

Figure 1: [Adapted from (Hofmann et al., 2020)]. Dynamic Contextualized Word Embeddings (dimensionality: 768). Each static embedding $\tilde{e}^{(k)}$ is mapped to a dynamic embedding $e_{ij}^{(k)}$ by some function d that incorporates time and social context. The points scattered around $e_{ij}^{(k)}$ are contextualized versions of $\tilde{e}^{(k)}$. Temporal and social space variability are captured by the variability in $\phi_{ij}^{(}k)$.

paper, we conduct several ablation studies beyond a replication of their model to answer two key questions:

1. Does BERT, which was the PLM used for the paper, perform better than other competing PLMs?

2. How does the performance of dynamic contextualized embeddings compare to dynamic non-contextual embeddings generated from techniques like Word2Vec?

To answer these questions, we introduce four models: GPT-2, RoBERTa, Word2Vec, and GLoVe. For each model, we conduct a downstream sentiment analysis task on embeddings created both with and without the dynamic component. Alongside our BERT baseline, we perform a series of comparisons that lend to better analysis of the contributions of each individual component of DCWEs.

2 Related Work

2.1 Overview of "Dynamic Contextualized Word Embeddings"

Our work builds upon the Dynamic Contextualized Word Embeddings technique (DCWE) proposed by Hofmann et al. (2020). This model uses a traditional contextual embedding as the base, with modifications to accommodate temporal and social context. Additional systems that contribute to the workings of DCWE are also discussed in later sections.

For DCWEs, Hofmann et al. (2020) follow a two stage system. Words are first injected with "dynamic" information based on the time and social context of the individual data point. They are then passed through a contextual embedding language model to alter the embeddings based on their immediate linguistic surrounding. The overview of the model is as follows.

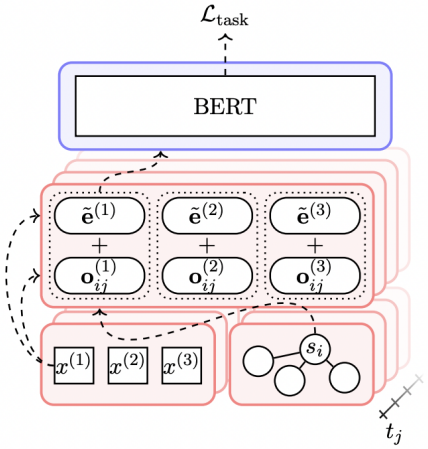

Figure 2: [Adapted from (Hofmann et al., 2020)]. The initial words x are used to generate embeddings $\tilde{e}$. The social and temporal context s_i and t_j (represented through Graph Attention Networks) are used to generate o_{ij}. Lastly, the two embeddings are concatenated and passed into the model, with the contextualizer's output being used to compute the task-specific loss.

With a set of words $[x^1, \ldots, x^n]$, the model derives a set of corresponding non-contextual embeddings $[\mathbf{e}^1, \ldots, \mathbf{e}^n]$. Since the model uses a PLM (BERT), the non-contextual embeddings are computed using the pretrained lookup table that maps words to embeddings. Let each stage of the model then be described by two functions, depicted in Equations 1 and 2.

$$d(x^k, s_i, t_j) = \mathbf{e}^k + \mathbf{o}_{ij}^k \tag{1}$$

This function combines the initial pretrained embedding with a vector $\mathbf{o}_{ij}^k$ that denotes the vector offset from the non-dynamic embedding that is specific to the unique s_i and t_j. However, the authors do make a modification to the $\mathbf{o}_{ij}^k$ term such that it follows the distribution $\mathbf{o}_{ij}^k \sim \mathcal{N}(\mathbf{0}, \lambda_a^{-1}\mathbf{I})$. This is done by adding a regularization term during the training step.

The second function is as follows:

$$\mathbf{h}_{ij}^k = \text{PLM}(\mathbf{e}_{ij}^k, E_{ij}^{<k}, E_{ij}^{>k}) \tag{2}$$

In this function, $\mathbf{h}_{ij}^k$ represents the final dynamic contextualized embeddings, $\mathbf{e}_{ij}^k$ represents the output of d, and E_{ij} represents the context before and after a given token. This equation represents the process of contextualizing the dynamically computed vectors. Using a PLM for this step was particularly advantageous to the authors, as it allowed for the model to be easily adapted for downstream tasks by simply adding one layer to the contextualizing model.

The final part of the model is the calculation of $\mathbf{o}_{ij}^k$ based on the values of s_i and t_j. This term was included in the above equations, but the calculation is as follows. The model creates a set of time-specific feed-forward networks and passes a vector created by concatenating $\mathbf{e}^k$ and $\mathbf{s}_{ij}$ through this network to get $\mathbf{o}_{ij}^k$. To compute the s_{ij} vector, the model represents the "community" of the author through a time-specific graph attention network (GAT) as proposed by Veličković et al. (2017) to encode the social community graph into a vector. Then, to model the temporal drift, the model imposes a random walk over $\mathbf{o}_{ij}^k$ such that $\mathbf{o}_{ij}^k \sim \mathcal{N}(\mathbf{o}_{ij}^k, \lambda_w^{-1}\mathbf{I})$, where $j' = j-1$ (Bamler and Mandt 2017, Rudolph and Blei 2018). The random walk prior ensures that the dynamic embeddings change smoothly over time.

The training process for this model is straightforward, since the inclusion of BERT as the PLM means only one task-specific layer needs to be added. Hofmann et al. (2020) use four datasets to train DCWEs: ArXiv, Ciao, Reddit, and Yelp. They run a series of tests to determine the effect of these dynamic components when compared to the normal contextual embeddings. They observe a masked language modeling perplexity that is similar, if not better, than non-dynamic contextualized word embeddings, indicating some success with the introduction of extralinguistic information. Furthermore, and particularly of note to this project, the authors note a uniform increase in performance that is statistically significant for the DCWE embeddings in a sentiment analysis task when compared to BERT embeddings with no dynamic component.

2.2 Related Work Regarding our Ablation Study

We aim to expand upon Hofmann et al. (2020) by changing the underlying model, with the goal of comparing contextual and non-contextual models. Traditional non-contextual models like GloVe (Pennington et al., 2014) and Word2Vec (Mikolov et al., 2013) compute *static* word embeddings, i.e., they represent each word with a single vector. As Hofmann et al. (2020) explain, this method of modeling semantics ignores the variability of word meaning across different contexts. On the other hand, contextual models incorporate linguistic context by mapping type-level representations to token-level representations (McCann et al., 2017). Contextual models like GPT-2 (Radford et al., 2019), BERT (Devlin et al., 2019), and RoBERTa (Liu et al., 2019) represent words as vectors that can vary across contexts. We thus choose to examine both contextual and non-contextual models to understand the importance of this contextual component.

Additionally, exploring the impact of model ablations on the *dynamic* component is helpful given the emergence of lexical semantic change detection in NLP research (Tahmasebi et al., 2018, Dubossarsky et al., 2019, Schlechtweg et al., 2017). Many methods on lexical semantic change detection also use static embeddings (Kim et al., 2014, Kulkarni et al., 2014), albeit with modeling disadvantages (Bamler and Mandt, 2017), so it is helpful for our ablation study to investigate them.

3 Statement of Purpose

Hofmann et al. (2020) propose a method to capture both temporal and social context as an additional layer on a pretrained BERT model and highlight potential application scenarios (masked language modeling and sentiment analysis) on four English datasets. In this paper, we focus on sentiment analysis. We reproduce the baseline results of DCWE on the Yelp dataset and present four model ablations: two contextual and two non-contextual.

4 Approach

Dataset The dataset choice for this project was constrained by the need to have samples tagged with their publication time and social context. In pursuit of this information, and because of the ease of transferring available pre-processing code, we choose to use the Yelp dataset. Yelp consists of 795,661 reviews. Critically, this dataset has explicit friendship relations that can be used to create a directed graph between users that is necessary to create the $\mathbf{s}_{ij}$ component. In order to perform binary sentiment analysis on this data, we modify the data such that one/two star ratings become neg-

ative, four/five star ratings become positive, and three star ratings are discarded. We randomly split this dataset into 70% train, 10% development, and 20% test. We present more information about the dataset and the directed graph in Table 1.

Experiments We conduct two kinds of experiments in this paper. The general pipeline for each type are listed below.

1. **Contextual Embeddings**
   (a) Extract pretrained embeddings from the contextualizing model's lookup table.
   (b) Use pretrained embeddings to compute the dynamic component.
   (c) Send pretrained embeddings injected with dynamic information into the contextualized model.
   (d) Train the sentiment analysis model with these embeddings to compare downstream task performance against paper baselines.
2. **Non-contextual Embeddings**
   (a) Train non-contextual embeddings on the entire corpus.
   (b) Apply the dynamic component to the non-contextual embeddings.
   (c) Train the sentiment analysis model with these embeddings to compare downstream task performance against paper baselines.

As described in section 2, the general pipeline of the DCWE model involves generating non-contextual embeddings, adding a temporal and social component to the embedding and passing that embedding through the contextualizing model (BERT in the paper). However, there are certain changes that have to be made in order to facilitate consideration of non-contextual embeddings. Namely, because unique embeddings are not created for each individual sample's context, we do not pass embeddings through a contextualizing layer as an intermediate step before downstream training.

5 Implementation

We build upon the codebase provided by Hofmann et al. (2020). The following section describes the pipeline followed by the existing codebase for the task of training DCWE on the Yelp Dataset for sentiment analysis.

Dataset Pre-processing We remove all duplicates, as well as all datapoints with fewer than ten words. Furthermore, because BERT can only accept 512-dimensional inputs, we truncate any reviews longer than that length by taking the first and last 256 words. We further tokenize the words for input into the PLMs.

Pretrained Language Models The many PLMs used throughout our paper are all implemented through the `transformers` library in Python. `transformers` provides each model, and also allows for initialization of the models based on the final pretrained weights, which were used in this project. Furthermore, since each model requires a different form of tokenization, we also use the included tokenizers within the library during our dataset pre-processing.

Embedding Calculation With the model selected through the `transformers` library, we access the internal lookup table for each model tested using the respective `get_input_embeddings()` function. This gives us the non-contextual embeddings $\mathbf{e}^k$. The social component is then calculated as described above and added to the non-contextual embedding vector. Finally, to obtain our embedding, we pass these dynamic embeddings into our model.

Sentiment Analysis Modifications Since we train on a sentiment analysis task, we add two additional fully connected layers to calculate a sentiment score. The first layer is a set of 100 fully connected nodes, which is passed through a $\tanh$ activation function and a 0.2 Dropout layer. The second layer is one fully connected node that is passed through a sigmoid activation function. This outputs a sentiment score, which is used to create the loss function that trains the network.

Models All of the models used in conducting the comparative studies are described in the section below. Notably, in addition to the baseline (BERT), we evaluate on Word2Vec, GloVe, GPT-2 and RoBERTa. In all of these settings, we analyze the effect on the downstream sentiment analysis task.

- **BERT:** Proposed by Devlin et al. (2019), BERT is a contextual transformer-based model, pretrained on the dual objectives of masked language modeling and next sentence

		Linguistic		Social						Temporal			
Dataset	$\|\mathcal{D}\|$	Unit	$\mu_{\|X\|}$	Unit	$\|\mathcal{S}\|$	$\|\epsilon\|$	μ_d	μ_π	ρ	Unit	$\|\mathcal{T}\|$	t_1	$t_{\|\mathcal{T}\|}$
YELP	795,661	Review	151.59	User	5,203	223,254	45.17	2.83	.009	Year	10	2010	2019

Table 1: Dataset statistics. $|\mathcal{D}|$: number of data points; $|X|$: average number of tokens per text; $|\mathcal{S}|$: number of nodes in network; $|E|$: number of edges; µd: average node degree; $_\pi$: average shortest path length between two nodes; ρ: network density; $|T|$: number of time points; t_1: first time point; $t_{|\mathcal{T}|}$: last time point.

prediction. We initialize it similarly to the paper, with the only modification being that we use DistilBeRT due to time constraints (Sanh et al., 2019).

- **Word2Vec:** Mikolov et al. (2013) propose Word2Vec as a two-layer neural network to learn word associations from text, where each word is represented by a unique vector. Notably, Word2Vec is trained on the entire corpus, creating global vectors, instead of unique contextual relationships. This requires some changes to the training loop, which are described above. We use the Gensim implementation (Řehůřek and Sojka, 2010).

- **GloVe:** Pennington et al. (2014) propose GloVe as an unsupervised algorithm to develop word representations. Training is performed using aggregated word co-occurence matrices. This is based on the assumption that co-occurrence probabilities encode meaning. Similar to Word2Vec, changes are made to the model in order to compare non-contextual embeddings. We use the Stanford NLP implementation.

- **RoBERTa:** Liu et al. (2019) present RoBERTa, a bidirectional transformers model that uses a masked language modeling training objective, similar to BERT. However, contrary to BERT, the masking occurs directly during pre-training, rather than during the training loop. RoBERTa is a contextual word embedding structure, which allows us to use the same setup as the original paper with a contextual model substitution (roberta-base).

- **GPT-2:** Radford et al. (2019) propose GPT-2 as a self-supervised transformers model that is trained using a causal language modeling objective (predicting the next word in sentences). This means that, contrary to models like BERT, GPT-2 is auto-regressive and therefore incorporates context from only the left of the token. However, this still yields contextualized embeddings, which allows us to use the same general structure as the paper, with a model substitution (distilgpt2) for GPT-2.

Technical Requirements All of the training above is completed using a GPU-accelerated instance of Colab Pro. We provide the training times in the results section, but most models take approximately 12 hours to complete 2 epochs.

6 Reproduction of Paper Baselines

To begin our investigation of the impact of contextualization on dynamic contextualized word embeddings, we first reproduce the baseline results of Hofmann et al. (2020). We provide a summary of the F1 scores across the contextualized word embeddings (CWEs) and dynamic contextualized word embeddings (DCWEs) on a sentiment analysis task below.

For reference, we run both CWEs and DCWEs for 2 epochs (compared to 2 epochs for CWEs and 3 epochs for DCWEs in Hofmann et al. (2020)). We use a learning rate of 3e-6, regularization constant (for DCWE only) of 1e-1, batch size of 4, and dimension of the social component of 768, consistent with Hofmann et al. (2020). The approximate training time is 648.1 minutes for CWEs, and 754.7 minutes for DCWEs.

	Paper		Us	
Model	Dev	Test	Dev	Test
DCWE	0.969	0.968	0.912	0.911
CWE	0.967	0.966	0.911	0.910

Table 2: F1 score on sentiment analysis for Yelp dataset (higher is better). DCWE: dynamic contextualized word embeddings; CWE: contextualized word embeddings.

We are able to reach a similar conclusion as

Hofmann et al. (2020), in that dynamic contextualized word embeddings achieve very slight, but significant improvements over the already strong performance of non-dynamic BERT. As expected, our accuracy on the Yelp dataset is slightly lower because we use distilBERT instead of $\text{BERT}_{\text{BASE}}$ (uncased) in the original paper, due to time constraints. We also train CWE for 2 epochs whereas the original paper trains for 3 epochs.

7 Ablations

We split this section into an analysis of contextualized models and an analysis of non-contextualized models. The details regarding our model training parameters can be found in the Appendix.

7.1 Contextualized Models

Hofmann et al. (2020) aim to build dynamic contextualized word embeddings that are based on the BERT PLM. We extend their work to understand if other PLMs achieve better performance. This is accomplished by changing the contextualizer to other pretrained models, namely GPT-2 and RoBERTa. Using the `transformers` library, we can exchange the pretrained model and follow the same process as was done for BERT. We report the results with and without the dynamic component for each contextualizer in Table 3.

Model	Dev	Test
BERT (DCWE)	0.912	0.911
BERT (CWE)	0.911	0.910
GPT-2 (DCWE)	0.505	0.508
GPT-2 (CWE)	0.517	0.518
RoBERTa (DCWE)	0.962	0.962
RoBERTa (CWE)	0.967	0.968

Table 3: F1 score on sentiment analysis for Yelp dataset across all contextual models (higher is better). DCWE: dynamic contextualized word embeddings; CWE: contextualized word embeddings.

The results seem to indicate that the dynamic component does not have a significant impact on the F1 score for non-BERT models. Additionally, GPT-2 performs substantially worse than the baseline both with and without the dynamic component. This may be due to the fact that RoBERTa and BERT are bidirectional encoders, whereas GPT-2 is unidirectional (only examines words that have already been seen in a sentence).

It is also evident that RoBERTa outperforms the baseline both with and without the dynamic component. Although the comparison is not direct since we use distilBERT instead of $\text{BERT}_{\text{BASE}}$, the result is still surprising, as distilBERT retains 97% of the performance (Sanh et al., 2019). This is a strong indication that RoBERTa may be a better choice for further analysis of the dynamic component.

7.2 Non-contextualized Models

We explore the impact of using models without a contextualizing component.

Model	Dev	Test
Word2Vec (DNWE)	0.464	0.463
Word2Vec (NWE)	0.461	0.461
GloVe (DNWE)	0.464	0.341
GloVe (NWE)	0.454	0.329

Table 4: F1 score on sentiment analysis for Yelp dataset across all non-contextual models (higher is better). DNWE: dynamic non-contextualized word embeddings; NWE: non-contextualized word embeddings.

The results do indicate some benefit being conferred by the dynamic component, in line with the contextual model results. However, the embeddings perform substantially worse than the baselines. We think this is because the pipeline we created to build non-contextual embeddings is reliant on tokenized data. The tokenization likely splits some words, which may interact poorly with global vectors trained on non-tokenized data. This presents an avenue for future work, where we could recreate the dataset to better interact with noncontextual embeddings.

8 Error Analysis

In addition to the F1 scores presented above, we conducted a sequence of error analyses to highlight the differences between the contextualizing models.

Methodology There are a sequence of additional steps to conduct the error analysis and generate the plots that follow. In the initial analysis, we train each of the three models. Then, using the test split of the dataset, we intercept the model before the dynamically computed embeddings are passed into the fully-connected layer. We compute an average embedding for the tokens over the entire test dataset for each of the three models. This

gives us a general representation of the embedding generated by each model. We identify tokens that were present in the reviews that RoBERTa, our best model, predicted correctly, but that BERT and GPT-2 failed to predict correctly. We then analyze the most frequent tokens in these reviews, excluding punctuation characters and stopwords. Finally, we conduct a two-component PCA to generate visualizations for each embedding in the model.

Results In total, there are 1,984 reviews that RoBERTA predicted correctly that GPT-2 and BERT predicted incorrectly. A substantial number of these reviews have two primary characteristics: either they are mixed leaning negative, or negate positive terms, e.g. "not good." Almost all of the reviews that are incorrectly predicted by the other two models were negative reviews. The ten most common tokens in this set after removing stopwords and punctuation are 'cat', 'text', 'feel', 'cell', 'mind', 'mini', 'water', 'point', 'people', and 'stud.' There does not appear to be a significant unifying factor or distinct "tone" in any of these tokens. This seems in line with the models failing on the less obviously positive or negative reviews.

To understand why RoBERTa potentially "understood" these reviews better, we conduct a principal component analysis (PCA) along two components to plot the embeddings for these tokens, alongside their five closest neighbors according to each model. These are contained below for the first two tokens ('cat' and 'text').

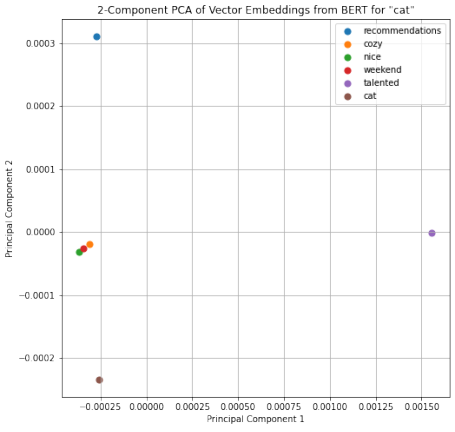

Figure 3: PCA for Bert "Cat"

PCA Analysis The GPT-2 and BERT embeddings seem to group words that are incredibly similar (e.g., text and message), while RoBERTa groups words that lack visible similarity. However, given

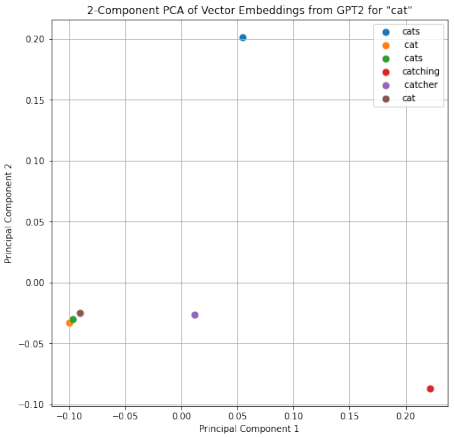

Figure 4: PCA for GPT-2 "Cat"

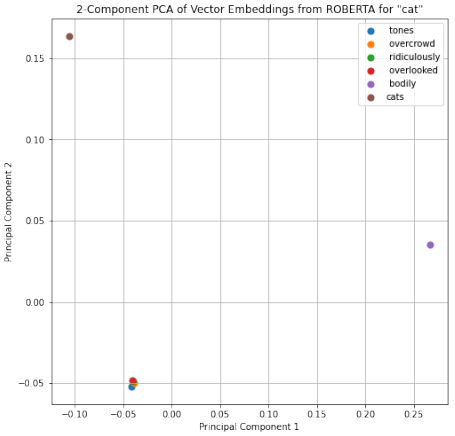

Figure 5: PCA for RoBERTa "Cat"

that RoBERTa seems to make better predictions based on these embeddings, it is possible that the model is able to make hidden connections between words based on their context, thus gaining additional "information." Therefore, the fact that the closer embeddings are less visibly related could be an advantage for RoBERTa.

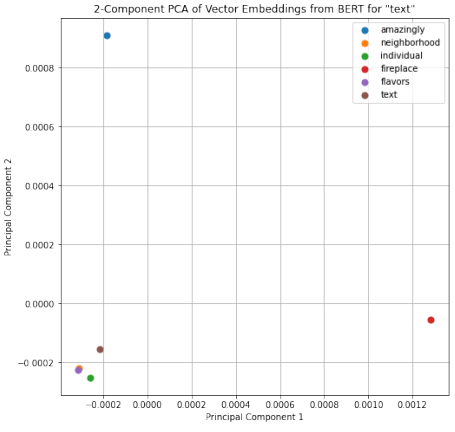

Figure 6: PCA for BERT "text"

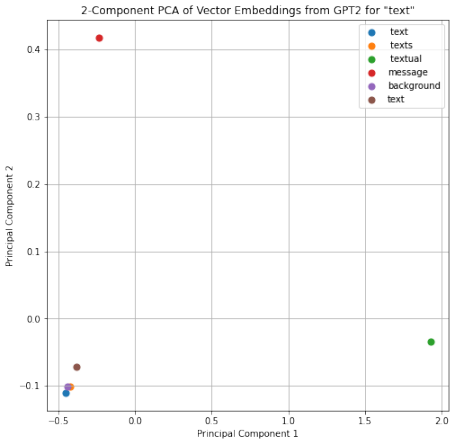

Figure 7: PCA for GPT-2 "text"

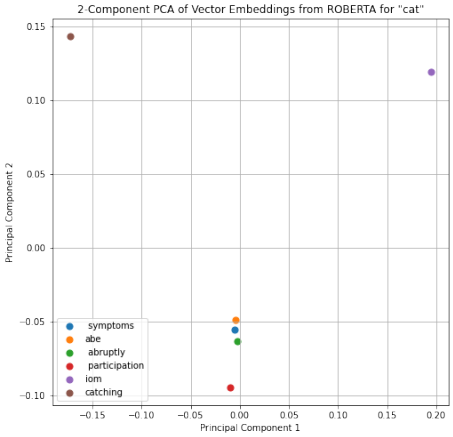

Figure 8: PCA for RoBERTa "text"

9 Conclusion

Modern NLP systems are often reliant on embeddings to represent meaning for tokens. However, many embeddings tend to lack context, both linguistic (surrounding text) and extralinguistic (time, community). These forms of context provide valuable information that can increase model performance for certain tasks.

In this paper, we investigate DCWE, a dynamic contextual embedding approach proposed by (Hofmann et al., 2020). We replicate the authors' baselines and find consistent results given our training limitations. Next, we perform ablation studies based on the underlying embedding model. DCWE was initially conceived using BERT, but we consider Word2Vec, GloVe, RoBERTa, and GPT-2 as other embedding models that can be injected with dynamic components. GPT-2 performs significantly below the baseline, though it does join the contextual models in performing better than the non-contextual embedding systems (which were substantially less performant, as expected). Furthermore, while GPT-2 likely suffers from being auto-regressive, and therefore being less able to incorporate context, we discover that RoBERTa performs substantially better than our baseline, justifying its future use in DCWE experimentation. Furthermore, not every model benefits from the dynamic component's inclusion. While our baseline results align with Hofmann et al. (2020) in that the dynamic embeddings from BERT perform better, the other two models do not benefit from the dynamic embeddings.

Future work can expand our ablation study to other datasets to determine whether the social and time components can be adequately captured in non-review contexts. Furthermore, we would like to extend the dynamic component/ablation analysis beyond sentiment analysis to determine if the value conferred by this component persists with other NLP tasks.

Acknowledgments

We would like to thank Professor Narasimhan for his helpful feedback and support throughout the class, as well as Valentin Hoffman for providing clarifications on the DCWE paper.

References

Robert Bamler and Stephan Mandt. 2017. Dynamic word embeddings. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 380–389. PMLR.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Haim Dubossarsky, Simon Hengchen, Nina Tahmasebi, and Dominik Schlechtweg. 2019. Time-out: Temporal referencing for robust modeling of lexical semantic change. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 457–470, Florence, Italy. Association for Computational Linguistics.

Valentin Hofmann, Janet B. Pierrehumbert, and Hinrich Schütze. 2020. Dynamic contextualized word embeddings.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. 2014. Temporal analysis of language through neural language models. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 61–65, Baltimore, MD, USA. Association for Computational Linguistics.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2014. Statistically significant detection of linguistic change.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA. `http://is.muni.cz/publication/884893/en`.

Maja Rudolph and David Blei. 2018. Dynamic embeddings for language evolution. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 1003–1011, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Dominik Schlechtweg, Stefanie Eckmann, Enrico Santus, Sabine Schulte im Walde, and Daniel Hole. 2017. German in flux: Detecting metaphoric change via word entropy. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 354–367, Vancouver, Canada. Association for Computational Linguistics.

Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to diachronic conceptual change. *CoRR*, abs/1811.06278.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2017. Graph attention networks.

Ziqian Zeng, Xin Liu, and Yangqiu Song. 2018. Biased random walk based social regularization for word embeddings. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 4560–4566. AAAI Press.

Appendix A: Embedding Training - Hyperparameters

Model	Dev	Test	n_e	b	l	λ_a	τ
BERT (DCWE)	0.912	0.911	2	4	3e-6	1e-1	754.7
BERT (CWE)	0.911	0.910	2	4	3e-6	-	648.1
GPT-2 (DCWE)	0.505	0.508	2	4	3e-6	1e-1	502.6
GPT-2 (CWE)	0.517	0.518	2	4	3e-6	-	665.6
RoBERTa (DCWE)	0.962	0.962	2	4	3e-6	1e-1	767.9
RoBERTa (CWE)	0.967	0.968	2	4	3e-6	-	704.8
Word2Vec (DNWE)	0.464	0.463	2	4	3e-6	1e-1	423.1
Word2Vec (NWE)	0.461	0.461	2	4	3e-6	-	379.4
GloVe (DNWE)	0.464	0.341	2	4	3e-6	1e-1	295.8
GloVe (NWE)	0.454	0.329	2	4	3e-6	-	264.2