M.S. THESIS

Integrating Neologisms to Pretrained Language Models: Using Assimilation-Inspired Embedding Synthesis

사전 훈련된 언어 모델에 신조어 통합: 동화에서 영감을 받은 임베딩 합성을 활용하여

2023 년 8 월

서울대학교 대학원

컴퓨터 공학부

정 소 영

M.S. THESIS

Integrating Neologisms to Pretrained Language Models: Using Assimilation-Inspired Embedding Synthesis

사전 훈련된 언어 모델에 신조어 통합: 동화에서 영감을 받은 임베딩 합성을 활용하여

2023 년 8 월

서울대학교 대학원

컴퓨터 공학부

정 소 영

Integrating Neologisms to Pretrained Language Models: Using Assimilation-Inspired Embedding Synthesis

사전 훈련된 언어 모델에 신조어 통합: 동화에서 영감을 받은 임베딩 합성을 활용하여

지도교수 전 병 곤

이 논문을 공학석사학위논문으로 제출함

2023 년 6 월

서울대학교 대학원

컴퓨터 공학부

정 소 영

정소영의 공학석사 학위논문을 인준함

2023 년 7 월

위 원 장	염 헌 영	(인)
부위원장	전 병 곤	(인)
위 원	김 진 수	(인)

Abstract

Recent research has shown that pretrained language models (PLMs) can become outdated over time and need adaptation to new words or concepts. While efficient approaches to adapt PLMs to new vocabularies have been studied in the fields of domain or cross-lingual adaptation, these methods have yet to be explored in a setting where vocabulary updates should occur timely, periodically, and on a small scale (adding 1s and 10s of new words with 1MB training data). Unfortunately, such methods either exhibit unsatisfactory performance or result in overfitting to the new vocabularies. Existing works in model editing of PLMs have also been tested to be ineffective for injecting unseen entities into PLMs. Our paper proposes a tailored method — W-SUM — for adapting PLMs to new vocabularies (i.e., neologisms) by mimicking how humans process their internal knowledge when encountering a new word or concept. Inspired by assimilation in Piaget's cognitive development theory, W-SUM leverages the rich knowledge inherent in embedding existing tokens of PLMs to find the optimal embedding of a new vocabulary through a weighted sum of existing token embeddings. We let the PLM find the optimal weight distribution via language modeling objective. We evaluate W-SUM on two language model probing tasks – ECBD and LAMBADA and validate W-SUM 's ability to acquire a good embedding for new vocabularies through semantic analysis.

Keywords: Pretrained Language Model, Temporal Adaptation, Neologism
Student Number: 2021-23818

Contents

Chapter 1

Introduction

Temporal adaptation of pretrained language models (PLMs) refers to adapting a PLM to new data that emerges after the model has finished the initial pre-training. Recent studies [1, 2, 3] have shown that PLMs without temporal adaptation degrades in model performance as time goes which highlights the necessity of regularly adapting PLMs. For example, a PLM trained with corpus upto 2018 shows performance degradation when tested with dataset of 2019 or 2020, and adapting the PLM with corpus generated in 2019/2020 mitigates the degradation. We denote such degradation as temporal degradation.

Accordingly, there has been active research on techniques for keeping PLMs current. One line of research works explores continuous post-training techniques for temporal adaptation of PLMs [1, 2]. Some works directly apply language modeling objectives to continuous post-training while some other works apply continual learning techniques [4, 5, 6]. Existing works have commonly diagnosed temporal degradation in settings where there is a relatively long period between updates (at least a quarter of a year) and an ample amount of corpora available

for the target period the LM is adapting (e.g., Twitter text from 2017 to 2018). We shift our focus towards more frequent and smaller-scale updates of PLMs in temporal adaptation.

One cause of temporal degradation is failed or suboptimal understanding of new words or entities unseen to PLMs during pretraining [1, 7]. In this paper, we mitigate temporal degradation by studying how to effectively and efficiently integrate neologisms (i.e., new words or entities) into PLMs. Previous works on injecting knowledges [8, 9] have obtained good results on editing some knowledge that a PLM already knows (e.g., the name of the current president in the United States), recent NLP tasks [10, 11] have found that these model editing methods are ineffective in understanding new entities that the PLM has never seen during pretraining (e.g., Aespa, AirTag). While existing works have focused on editing of knowledge in PLMs, we aim to make a tailored method for addition of knowledge to PLMs.

This paper provides a new method – W-SUM – for integrating neologisms into PLMs, by mimicking how humans process their knowledge when encountering a new word or concept. Humans try to understand the new word based on the prior they have accumulated with past learning experiences (assimilation in Piaget's cognitive development). Likewise, W-SUM finds the optimal embedding of new vocabulary by leveraging the rich knowledge inherent in the embedding of existing tokens of PLMs. We treat a neologism as a single token and add the new single token to a PLM. Here we were influenced by vocabulary expansion approaches [12, 13, 14] studied in PLM's domain and cross-linguistic adaptation. The embedding of the new token is represented as a weighted sum of embeddings of the existing tokens, and we let the model find the optimal weight distribution via language modeling objective. Our training scheme is based on language modeling with unlabeled corpus, so our method does not

require data annotation.

Adding a new token to PLMs should meet two requirements: to learn the new token effectively and not to affect the meaning of existing tokens. We evaluate W-SUM on two language model probing tasks – ECBD and LAMBADA – and validate how well the method can meet both desiderates. We also conduct qualitative analysis on embeddings of new tokens found by W-SUM and demonstrate that W-SUM truly enables the PLM to find good embedding values for new tokens.

Chapter 2

Background

2.1 Temporal Adaptation of Pretrained Language Models

The temporal misalignment of pretrained language models (PLMs) has recently received much attention from the NLP research community. This issue refers to PLMs performing poorly on test data that originated after the models finished pretraining. Consequently, there have been various approaches for keeping PLMs up to date, namely, temporal adaptation. A temporal adaptation of pretrained language models (PLMs) refers to adapting a PLM to new data that emerges after the model has finished the initial pretraining. In this section, we look into two main approaches in temporal adaptation – continuous post-training and model editing.

2.1.1 Continuous Post-training of PLMs

Continuous post-training of PLMs refers to continuing to pretrain language models that have already finished pretraining on a large general corpus with the additional dataset. Existing works have taken two approaches when applying continuous post-training to temporal adaptation. The first approach is to continuously post-train PLMs with the same language modeling objective on recent corpus [1, 3, 2]. The second approach is to apply continual learning techniques to continuous post-training [5, 6, 4].

2.1.2 Model Editing on PLMs

Model editing refers to editing a knowledge of PLM acquired during pretraining. This line of works [8, 9] aims to make a more fine-grained, targeted update to PLMs compared to continuous post-training. Also, they focus on editing the existing face inside PLMs, not on appending some new word or entity that the language model has never seen during pretraining. As such, it has recently been diagnosed that these model-editing techniques are ineffective for injecting a piece of new information into a PLM [10], performing worse than the original PLM. Therefore, there has yet to be a method tailored for integrating a PLM to understand a new entity or neologism that the PLM has never seen.

2.2 Vocabulary Expansion in Domain Adaptation and Cross-Lingual Adaptation

Adapting a PLM to handle new vocabularies has been actively studied in domain adaptation (DA) and cross-lingual adaptation (CLA) of PLMs. DA refers to adapting a language model, which has been pre-trained on a large amount of general corpus, to a specific domain such as Bio, Computer-Science, etc., with

the associated domain-specific corpus. CLA aims to adapt a language model, which has been pretrained on a large amount of corpus in language A to some other language B. In most cases, language models pretrained over the English corpus get transferred to other mid-resource/low-resource languages.

One of the critical challenges in the DA/CLA is how to effectively capture the semantics of domain-specific / language-specific terms, which are out-of-vocabulary for the original PLMs' tokenizers. One primary approach frequently taken in DA and CLA is to expand PLM's embedding layer and tokenizer with new vocabularies. Research works propose different methods to initialize the embedding of these new vocabularies. They can be categorized as the following – 1) to leverage the overlap between the original corpus and the target domain corpus [15, 16] and 2) to continuously post-train with language modeling objective [12, 13, 14].

Our method follows the vocab expansion approach but enables more informed initialization by leveraging rich semantic information innate in PLM's existing token embeddings. We did not consider the first line of approach as it is more realistic to assume that the source corpus of PLMs is unavailable for end-users. The first line of work first learns the mapping between source and target vocabularies and then acquires the embeddings for target vocabularies using the trained mapping.

Chapter 3

Method

3.1 Main Idea

The main idea is to represent the embedding of a target token as the weighted sum of existing vocab embeddings, where we let the language modeling objective decide the weight for each vocab. This method was inspired by assimilation in Piaget's theory of cognitive development, which indicates that humans learn new information based on what they've acquired in the past. From this postulation, we devised this new method where the embedding value of a new vocabulary is deduced from the embedding value of pre-existing, well-defined tokens in PLM's vocab set. The optimal weight distribution is found via language modeling objective when the PLM is trained on an unlabeled corpus containing the new word.

Figure 4.1 illustrates the overall training process. We target single-word neologisms and extend a PLM's vocabulary with the neologism as a new token 'COVID-19' is added in the figure. Given V and N where V is the number of

vocabularies for the PLM and N is the number of new tokens to add, W-SUM optimizes a weight parameter of size N x V. In the figure, the weight parameter is illustrated as a single layer with size of 1 x V as we are only adding one token (COVID-19).

Then, we further pretrain the PLM with the language modeling objective on a small corpus that includes the newly added word. However, instead of updating the whole model parameter (Figure 3.2 (a)) or only the embedding layer (Figure 3.2 (c)), W-SUM updates the weight parameter only. Then the embedding for the new token (1 x H) is obtained by matrix-multiplying the weight parameter (1 x V) and the embedding layer (V x H) where H is the PLM's hidden dimension.

For the training dataset, W-SUM requires a corpus that includes the newly added token. Regarding the size of the corpus, we used a dataset of size 100s for evaluation (250 examples for the ECBD task and 500 examples for the LAMBADA task), which is reasonably small compared to the amount of corpus used for continuous post-training. We assume that this requirement is not challenging nor unrealistic as there would be enough mentions of the neologism if it holds enough importance to get integrated into PLMs. Also, as we are using an unlabeled corpus, it is free of burden for annotation.

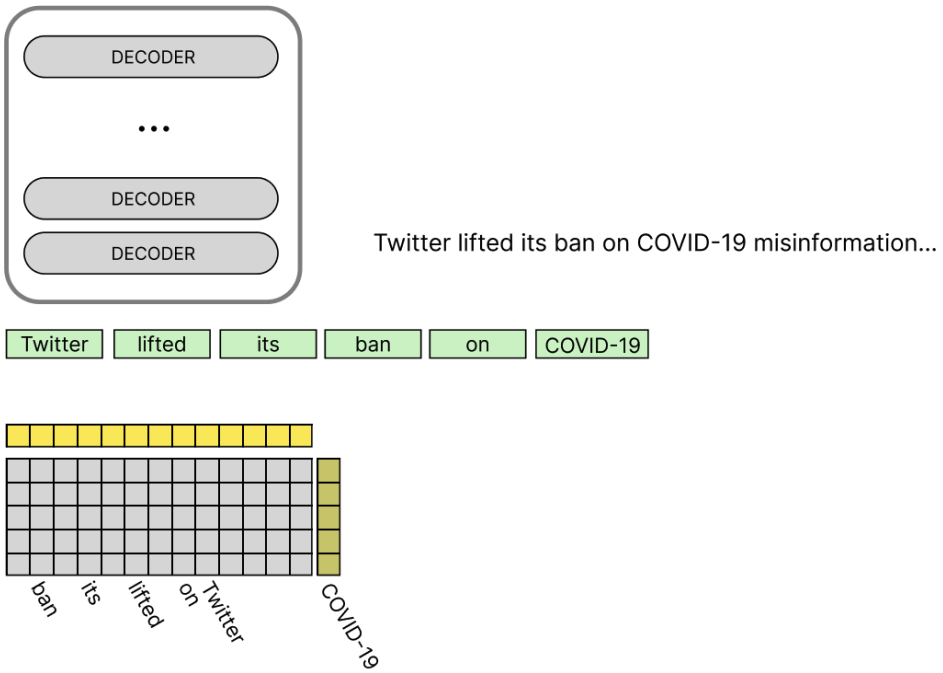

Figure 3.1 Diagram for W-SUM . A single layer of parameters colored in yellow refers to the weights per existing embedding tokens tuned by the language modeling objective. As the number of tokens to add increases, the height of the weight parameter (i.e., the number of layers) will grow linearly.

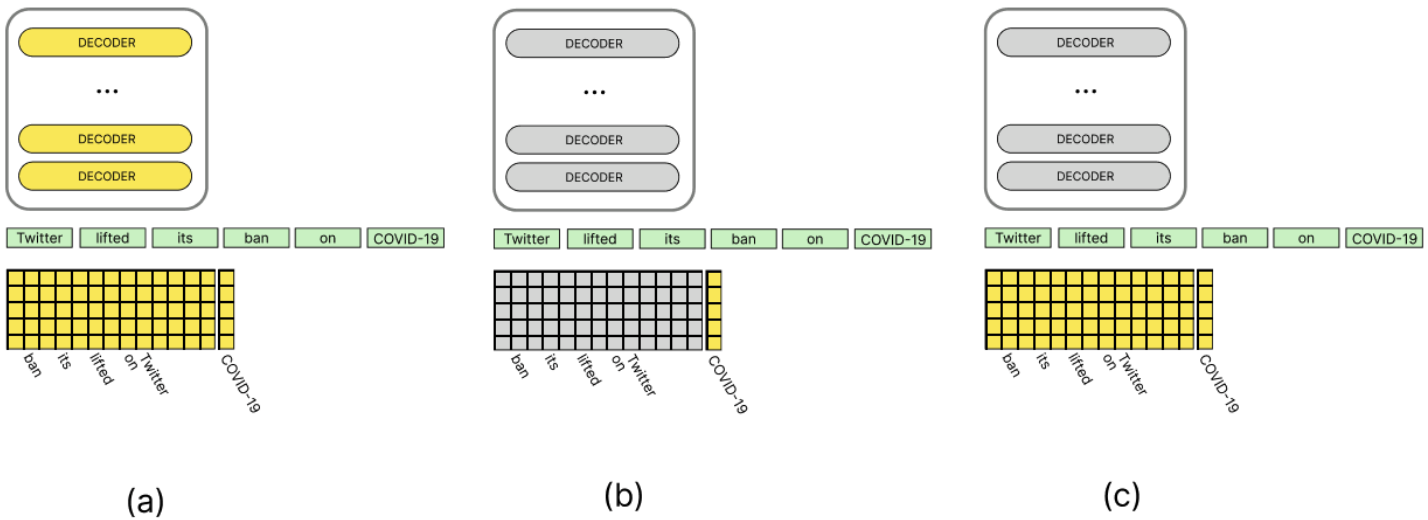

Figure 3.2 Diagram for baseline techniques.

Chapter 4

Evaluation

4.1 Evaluation Setup

In this section, we explain evaluation tasks, baselines compared with W-SUM , and the environment used for implementation and experiments.

4.1.1 Evaluation Tasks

W-SUM and baseline methods are evaluated on two NLP tasks – Entity Cloze By Date (ECBD) and LAMBADA.

Entity Cloze By Date

ECBD [11] is an entity probing task that tests whether a language model can infer about entities not seen during pretraining. The task provides a set of entities and their corresponding years when they have originated. The task also includes the entitie's definition and probe sentences with masked spans. Then, given definitions about a target entity, a PLM is tested whether it can make a

good guess on the masked span of the probe sentence.

We select entities to experiment with from the period of 2020 and 2021 to ensure that PLMs have not seen those entities during pretraining. Within the ECBD test dataset, 53 and 13 new entities for 2020 and 2021 each consist of a single word. As W-SUM requires some amount of corpus containing the target token to add newly, we picked entities with the most number of definitions provided. Such picked entities are *GPT-3*, which is a kind of PLMs that recently became famous, *Zynn*, which is a start-up company, and AirTag, which is a newly launched Apple product. We then applied input augmentation to the definitions to prepare a train dataset of around 250. For the augmentation, we prompted the ChatGPT [17] to paraphrase given definitions. We have rigorously checked that the paraphrased sentences do not add new information other than what is given by the original definitions; thus, other input augmentation techniques will also work fine.

LAMBADA

LAMBADA [18] is a natural language understanding task where a model is given a paragraph with the last few phrases masked, and asked to predict the masked portions. The given paragraphs are constructed so that the model should understand the whole context of the paragraph, not just the individual sentences close to the masked portions.

From the LAMBADA test set, we select only the test examples where the masked portion consists of a single token when tokenized, our method is designed to add a single new token to a PLM. Such test examples result in 1786 unique target tokens and 2819 test examples containing the target tokens. As training with 1786 unique tokens is resource-consuming, we randomly pick 216 tokens from 1786 tokens, denote them as LAMBADA-216, and mainly run ex-

periments on these examples. LAMBADA-216 includes 225 test examples. We believe the result averaged over 216 tokens is generalized enough to conclude how well our newly proposed method works.

Although the last tokens in LAMBADA examples are not really neologisms to PLMs, as models have seen them during pretraining, we still test with these data samples as temporal misalignment of PLMs is a relatively new research topic and there are few datasets available that provide real neologisms and corresponding test examples to evaluate. When testing with LAMBADA, we erase the true embedding of a given token from a PLM and re-learn its embedding using our method and baseline methods, to test whether they can successfully recover the given token's true embedding. We denote such tokens from LAMBADA examples as PSEUDO-NEOs. Also, we denote tokens that were truly unseen during pretraining to be REAL-NEOs.

In order to measure the specificity performance, we set aside a few test examples that do not overlap with LAMBADA-216. We randomly picked 150 unique tokens from the 1786 tokens not included in LAMBADA-216. They are 223 test examples, roughly matching the test examples number of LAMBADA-216 (225 examples).

4.1.2 Baselines and Models

We compare W-SUM with two methods: train-embed-wholeand train-embed-new. train-embed-whole(Figure 3.2 (c)) refers to augmenting PLM's vocabulary and embedding parameter with a new neologism and training the entire embedding parameter with a dataset on the neologism. This method was shown to be effective in adapting PLMs to a new domain [13]. train-embed-new(Figure 3.2 (b)) refers to augmenting PLM's vocabulary, embedding parameters with a new neologism, and training the parameter for the newly added token only.

This approach has also been tested in the domain adaptation of PLM [14]. In the original work, this method was applied to a PLM mixed with the original token embedding in the original work. Still, we use only the newly acquired embedding as the original embedding is unavailable in our setting, where we aim to find the embedding for unseen entities. When comparing our method with train-embed-new, we can also observe whether not only initializing but also modeling with the weighted sum of existing tokens truly helps to learn the embedding of a new token.

We did not add model editing approaches [9, 8] as they were shown to be ineffective for adding new entities to PLMs resulting in worse perplexity than the original PLM [10]. For ECBD task, we additinally report the performance of prepending definitions following the setting of the original work [10]. However, we do not add it as a baseline method for the LAMBADA task as it is not considered a valid method for injecting a new entity to a PLM [10] and it is not so straightforward how to apply it to the LAMBADA task.

We conduct all the experiments on GPT2-large [19].

4.1.3 Environment and Frameworks

All the implementation was done with PyTorch 2.1.0, and checkpoints for base models were downloaded from Huggingface. We evaluated W-SUM and baselines on NVIDIA A100 80GB GPUs for the LAMBADA task and NVIDIA A6000 40GB GPUs for the ECBD task. All experiments were run with tf32 mode activated for the LAMBADA task to accelerate the training progress, as the LAMBADA dataset is huge in size (1786 target tokens and 2819 evaluation examples in total).

4.2 Main Results

4.2.1 Result on ECBD

Method	GPT-3		Zynn		AirTag	
	Target	Spec	Target	Spec	Target	Spec
base model	65.83	39.76	40.54	39.76	51.59	70.54
train-embed-new	72.48	40.27	45.36	41.36	51.03	71.10
train-embed-whole (early)	69.28	40.12	36.93	40.08	48.44	70.02
train-embed-whole (late)	75.17	74.29	20.49	81.08	42.26	108.48
train-weight (Ours)	**38.47**	39.82	**30.28**	40.38	**45.78**	68.70
prepend-defs-full	18.11	*39.76*	14.02	*39.75*	29.41	*70.54*
prepend-defs-half	23.60	*39.76*	22.79	*39.75*	63.82	*70.54*

Table 4.1 Target perplexity and specificity perplexity measured on a subset of ECBD. The lower the perplexity is, the better. The best perplexity scores among all the methods are marked with underline and the best perplexity scores within parameter update based methods are marked as bold. The perplexity of train-embed-wholeat later training steps was not considered as its specificity perplexity is too high to be used.

Table 4.1 shows W-SUM and baseline methods' perplexity on ECBD task. W-SUM achieves lower target perplexity than the base model while maintaining the specificity perplexity same. This is better achievement compared to model editing techniques [9, 8] studied in the work of PLM learning new entities [10] where they resulted in even higher perplexity than the base model. W-SUM also consistently achieves lower perplexity compared to train-embed-new. While train-embed-wholeresults in lower perplexity than W-SUM at the later training

Method	Target Accuracy	Specificity Accuracy
base model	43.111	43.103
train-embed-new	28.000	37.570
train-embed-whole	N/A	N/A
train-weight (Ours)	64.444	24.472

Table 4.2 Target accuracy and specificityy accuracy measured on LAMBADA-216. We did not measure the score of train-embed-whole on the full LAMBADA-216, as the method consistently exhibits poor performance on the smaller subset (Figure 4.1).

steps, the embeddings from train-embed-wholecannot be used as they show steep surge in specificity perplexity. We conjecture that updating the whole embedding parameter results in overfitting, thus, badly impacting the overall next token prediction ability.

While W-SUM achieves comparatively good performance among techniques based on parameter update, its perplexity is still higher than that of prepending definitions during inference. Although prepending definitions during inference was not considered a valid solution for adapting PLMs because of its inference cost [10], there is room for improvement in parameter update-based PLM adaptation techniques considering the reasonably low perplexity achieved by prepending definitions.

4.2.2 Result on LAMBADA

Table 4.2 shows the target accuracy and specificity accuracy meausred on LAMBADA-216. We trained each method for 75 steps, and evaluated against LAMBADA-216 for every 15 steps. Please note that the score for train-embed-

whole is not measured as the method showed poor performance (0 target accuracy) when evaluated on a smaller subset of LAMBADA-216. In Figure 4.1, it is observed that train-embed-whole scores 0 target accuracy all for five evaluations, averaged over 30 tokens randomly selected from LAMBADA-216. We conjecture that the parameter update on the entire embedding layer may have caused too much distribution shift from the original PLM that the model cannot correctly perform the next token prediction task.

The result implies that train-weight achieves increased of 21.333p% in target accuracy compared to the base model. Meanwhile, train-embed-new exhibts decrease in target accuracy, resulting in 28.0%. This implies that applying weighted sum modeling when training the newly added embedding token helps the PLM to find the embedding value for the new token.

However, this accuracy gain comes at the cost of specificity accuracy. Looking at the specificity accuracy of train-embed-new, adding a newly trained embedding token alone badly impact the specificity accuracy by showing a 5.533p% decrease. Applying weighted sum modeling further degrades the specificity accuracy by 13.099p%. This result contrasts with the evaluation result we saw in the ECBD task. In the evaluation result of the ECBD task, neither train-weight nor train-embed-new showed a decrease in specificity perplexity compared to the original PLM. Such a decrease in specificity accuracy in the LAMBADA task may be exaggerated. The LAMBADA task, a next token prediction task, is essentially a classification task over 50K classes (equivalent to the number of); thus, even a slight change in the embedding layer may badly impact the classification result. We believe it is more appropriate to choose the perplexity metric, as we did in the ECBD task, to quantify the amount of possible harm brought to other existing tokens.

We also evaluated train-weight and train-embed-new with the whole LAM-

BADA task on 1786 unique tokens. In Figure 4.2, we can observe that with train-weight the target accuracy continues to improve as training progresses while train-embed-new does not.

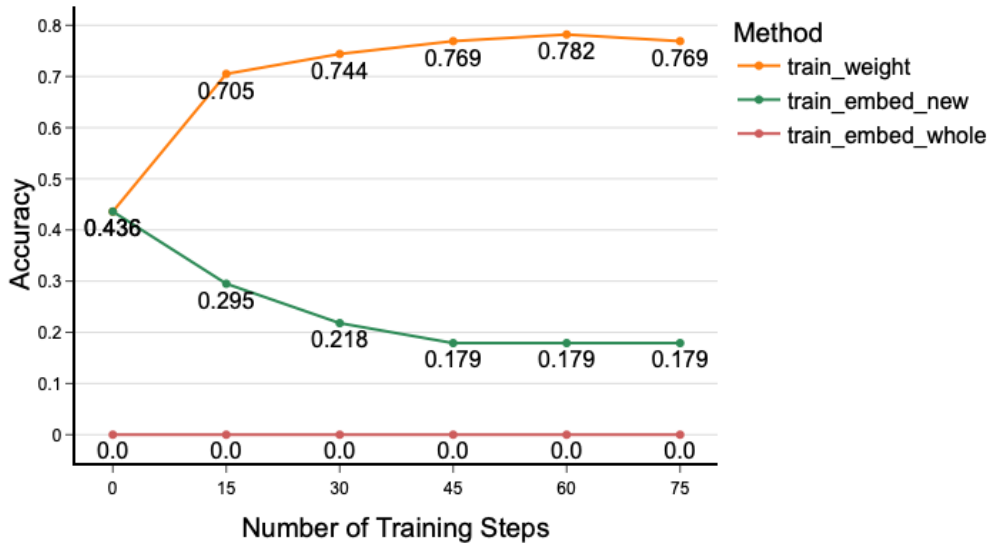

Figure 4.1 Evaluation on a smaller subset of LAMBADA-216, averaged over 30 tokens randomly picked.

4.3 Analysis on the Semantics of Token Embeddings

This section discusses how well W-SUM and baseline methods acquire the appropriate semantics of the target token. In detail, we first investigate the relatedness between the target token embedding trained and the original token embedding. The closer they are, the better trained the target token embedding is. Also, we present examples of which token embeddings of the original PLM are the closest to the target token. For measuring the level of closeness, we use

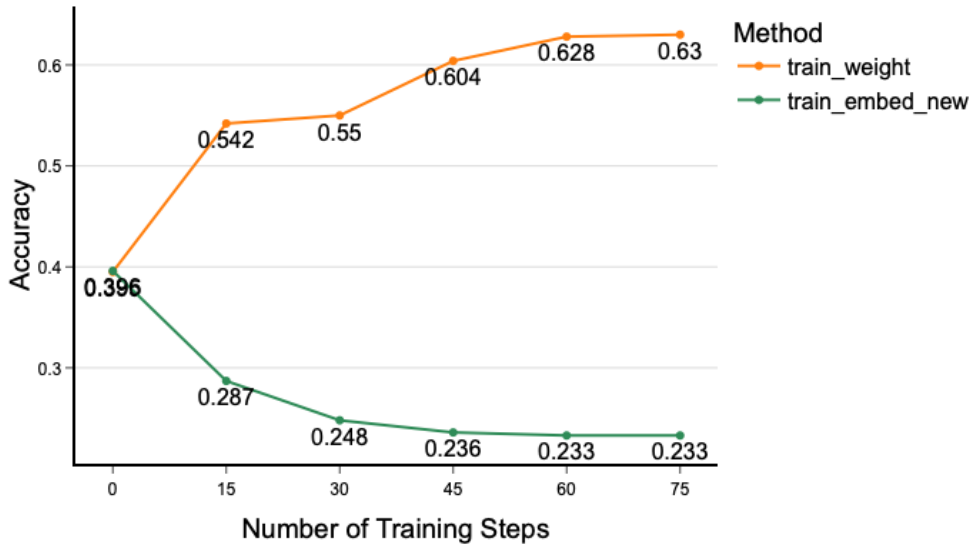

Figure 4.2 Evaluation on the whole LAMBADA task, averaged over 1786 tokens. The score for train-embed-whole is not measured as the method exhibits poor performance (target accuracy of 0) when tested on a smaller subset of LAMBADA-216.

cosine similarity as it is one of the most widely used metrics used to measure how similar two word embeddings are.

4.3.1 Evaluation with PSEUDO-NEO

Figure 4.3 shows how the cosine similarity between the embedding acquired from training and the oracle embedding (i.e., the original embedding of the target token) changes as training progresses. The target tokens are 216 tokens from LAMBADA-216, and the cosine similarity was measured for every 15 training steps, six times in total. The result in the figure is the average over

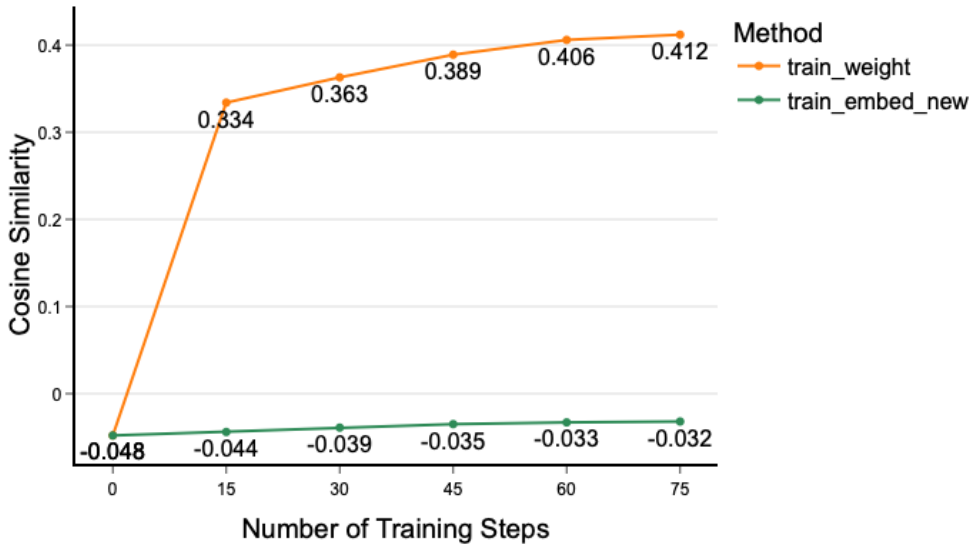

Figure 4.3 The trend of cosine similarity with the original embedding as trainig progresses.

216 tokens. The labels of the x-axis of the figure represent the step at which the embedding of the target token was evaluated against the oracle embedding. Please note that the value at step 0 refers to the cosine similarity before training, that is, the cosine similarity between the initialized and oracle embedding. The baseline train-embed-wholeis not included as it demonstrated noticeably poor accuracy for the LAMBADA task during the preliminary experiment (shown in Figure 4.1).

W-SUM demonstrates a steady increase in cosine similarity with the oracle embedding throughout training, and at the last step, it reaches the cosine similarity of 0.412. On the other hand, the baseline train-embed-newfails to go over the cosine similarity of 0 after 75 training steps, although the value at the last

step (-0.032) has increased slightly compared to the beginning of the training (-0.048). This result implies that W-SUM finds semantically better embedding than the baseline when given the same amount of training.

In Table 4.3, we present the 10 most similar tokens to the trained embedding per each target token. We searched the most similar tokens based on the last step's checkpoint both for W-SUM and train-embed-new.

Similar embedding tokens found for W-SUM seem to contain related words to the original target token. (e.g. Ġbolstered or Ġrevived for Ġrenewed) For some cases, they also include seemingly unrelated tokens to the original target token, such as ÃĥÃĤÃĥÃĤÃĥÃĤÃĥÃĤÃĥÃĤ or ʁļåŁ« in the case of Ġbelief. What caused the embedding to be located near seemingly unrelated tokens during the training of W-SUM is one interesting future research agenda.

Another intersting observation is that W-SUM seems to recover the contextualized meaning of tokens that possess different meanings in various contexts. For example, considering Ġclass, the closest embedding tokens found by W-SUM include Ġproletariat, Ġlineage and Ġcohort, which are synonyms for the word "class" within the context of socialism, genetics, and sociology, respectively. This implies that learning embeddings by W-SUM reflects the contextualization process language modeling.

4.3.2 Evaluation on REAL-NEO

Unlike PSEUDO-NEOs, with REAL-NEOs, measuring the cosine similarity with the original embedding token is inappropriate as the target token is entirely new to the PLM, and the original embedding token does not exist. Instead, examples of the most similar tokens from the PLM's embedding space are presented in Table 4.4. Similar tokens in the table show that W-SUM retrieves semantically related tokens, while the other two methods do not.

Target Token	Method	Top-10 Cosine Similar Embedding Tokens
Ġrenewed	train-weight	Ġrenewed, Ġheightened, Ġbolstered, Ġreiterated, Ġintensified, Ġrenovated, Ġstrengthened, Ġrevived, Ġrevamped, Ġreopened
	train-embed-new	Ġin, Ġ(, Ġthe, Ġused, -, Ġand, „ Ġis, Ġa, Ġ"
Ġmistaken	train-weight	Ġunintention, Ġmisguided, Ġnonsensical, Ġpractition, Ġsubur, Ġerroneous, Ġmistaken, Ġdismissive, Ġdelusional, Ġmisinterpret
	train-embed-new	Ġin, Ġ(, Ġused, -, Ġthe, Ġand, Ġis, „ Ġuse, Ġa
Ġclass	train-weight	Ġproletariat, Ġpercentile, Ġcaste, Ġteasp, Ġslab, Ġsubur, Ġcarbohyd, Ġpione, Ġcohort, Ġlineage
	train-embed-new	Ġin, -, Ġ(, Ġand, Ġis, Ġthe, Ġused, „ Ġuse, Ġ"
Ġsleeve	train-weight	Ġsleeve, Ġsleeves, Ġjacket, Ġskillet, Ġskirt, Ġlace, Ġholster, Ġsweater, Ġjackets, Ġnecklace
	train-embed-new	Ġin, Ġ(, -, Ġand, Ġused, Ġthe, Ġis, „ Ġuse, Ġ"
Ġbelief	train-weight	Ġbelief, Ġreverence, Ġpractition, Ġunintention, Ġtheological, Ġpione, DeliveryDate, ʞḷåℒ«, ÃĥÃĤÃĥÃĤÃĥÃĤÃĥÃĤ, Ġskepticism
	train-embed-new	Ġin, Ġ(, -, Ġand, Ġused, Ġis, Ġthe, „ Ġuse, Ġ"

Table 4.3 Examples of Top-10 cosine similar embedding tokens found. The target token is marked with an underline if found in the top 10 most similar embedding tokens.

Target Token	Method	Top-10 Cosine Similar Embedding Tokens
ĠAirTag	train-weight	ĠiCloud, ĠChromebook, ĠMacBook, ĠKinect, Ġepigen, ĠiPod, ĠAlibaba, ĠMessi, ĠXiaomi, ĠEminem
	train-embed-new	Ġa, Ġthe, Ġuse, Ġan, Ġfood, Ġproduction, Ġwork, Ġdevelopment, Ġtechnique, Ġmore
	train-embed-whole	ĠAirTag, Ġthe, ĠU, ĠApple, Ġ20, Ġ14, Ġ11, Ġ2, Ġit, Ġpre
GPT-3	train-weight	LinkedIn, Austral, Analy, JSON, NVIDIA, Google, Interestingly, Researchers, ß, Parameters
	train-embed-new	Ġfood, Ġsnacks, Ġhighly, Ġgimm, Ġdevelopment, Ġtechnique, Ġprotein, Ġproduction, Ġfoods, Ġproductive
	train-embed-whole	GPT-3, „ ’s, ., Ġand, -, Ġover, Ġin, Ġis, Ġpre
ĠZynn	train-weight	ĠYelp, ĠXiaomi, ĠHuawei, ĠHulu, ĠRoku, ĠDropbox, Ġcryptocurrency, ĠSpaceX, ĠLyft, ĠAirbnb
	train-embed-new	Ġa, Ġthe, Ġan, Ġmore, Ġfood, Ġall, Ġit, Ġhigh, Ġproduction, Ġthis
	train-embed-whole	ĠZynn, Ġthe, Ġin, Ġa, Ġ20, ĠAmerica, ., Ġall, Ġit

Table 4.4 Examples of Top-10 cosine similar embedding tokens found using each method. Please note that target tokens are REAL-NEOs, that is, a PLM has never seen these target tokens during pretraining.

Chapter 5

Discussion and Conclusion

In this paper, we have studied how to integrate neologisms into pretrained language models (PLMs) effectively. While there has been active research on the temporal degradation of PLMs and adaptation techniques to overcome the degradation, there has yet to be a technique tailored for integrating new words or entities, unseen during pretraining, into PLMs. This work proposes a new training method, W-SUM , inspired by how human intelligence assimilates new concepts into its internal knowledge model based on prior knowledge. Evaluations on language model probing tasks and qualitative analysis have validated that W-SUM does help to find good embeddings for newly added tokens to PLMs.

However, there are limitations, mainly regarding W-SUM 's applicability. First, the current form of W-SUM only applies to a single word's neologisms. While our work did not target multi-word neologisms as the understanding of multi-word expressions fall under an independent research field; it would be desirable for the technique to support multi-word neologisms as well as many

neologisms are phrases. Also, W-SUM do not assume cases where existing tokens of a PLM have acquired a new meaning but there exist cases where existing tokens should be added with new contexts. For example, the token 'ĠDelta' already exists in GPT2 models but still needs adaptation as the word 'Delta' in a new corpus is often associated with COVID-19. It would be advantageous for the technique to accommodate such cases as well.

Bibliography

[1] A. Lazaridou, A. Kuncoro, E. Gribovskaya, D. Agrawal, A. Liska, T. Terzi, M. Gimenez, C. de Masson d'Autume, T. Kočiský, S. Ruder, D. Yogatama, K. Cao, S. Young, and P. Blunsom, "Mind the gap: Assessing temporal generalization in neural language models," in *Advances in Neural Information Processing Systems* (A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), 2021.

[2] D. Loureiro, F. Barbieri, L. Neves, L. Espinosa Anke, and J. Camacho-collados, "TimeLMs: Diachronic language models from Twitter," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, (Dublin, Ireland), pp. 251–260, Association for Computational Linguistics, May 2022.

[3] K. Luu, D. Khashabi, S. Gururangan, K. Mandyam, and N. A. Smith, "Time waits for no one! analysis and challenges of temporal misalignment," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Seattle, United States), pp. 5944–5958, Association for Computational Linguistics, July 2022.

[4] X. Jin, D. Zhang, H. Zhu, W. Xiao, S.-W. Li, X. Wei, A. Arnold, and X. Ren, "Lifelong pretraining: Continually adapting language models to emerging corpora," in *Proceedings of BigScience Episode #5 – Workshop on Challenges & Perspectives in Creating Large Language Models*, (virtual+Dublin), pp. 1–16, Association for Computational Linguistics, May 2022.

[5] J. Jang, S. Ye, C. Lee, S. Yang, J. Shin, J. Han, G. Kim, and M. Seo, "TemporalWiki: A lifelong benchmark for training and evaluating ever-evolving language models," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, (Abu Dhabi, United Arab Emirates), pp. 6237–6250, Association for Computational Linguistics, Dec. 2022.

[6] J. Jang, S. Ye, S. Yang, J. Shin, J. Han, G. KIM, S. J. Choi, and M. Seo, "Towards continual knowledge learning of language models," in *International Conference on Learning Representations*, 2022.

[7] S. Amba Hombaiah, T. Chen, M. Zhang, M. Bendersky, and M. Najork, "Dynamic language models for continuously evolving content," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery; Data Mining*, KDD '21, (New York, NY, USA), p. 2514–2524, Association for Computing Machinery, 2021.

[8] E. Mitchell, C. Lin, A. Bosselut, C. Finn, and C. D. Manning, "Fast model editing at scale," in *International Conference on Learning Representations*, 2022.

[9] K. Meng, D. Bau, A. J. Andonian, and Y. Belinkov, "Locating and editing factual associations in GPT," in *Advances in Neural Information Processing Systems* (A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, eds.), 2022.

[10] Y. Onoe, M. J. Q. Zhang, S. Padmanabhan, G. Durrett, and E. Choi, "Can lms learn new entities from descriptions? challenges in propagating injected knowledge," 2023.

[11] Y. Onoe, M. Zhang, E. Choi, and G. Durrett, "Entity cloze by date: What LMs know about unseen entities," in *Findings of the Association for Computational Linguistics: NAACL 2022*, (Seattle, United States), pp. 693–702, Association for Computational Linguistics, July 2022.

[12] M. Artetxe, S. Ruder, and D. Yogatama, "On the cross-lingual transferability of monolingual representations," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, (Online), pp. 4623–4637, Association for Computational Linguistics, July 2020.

[13] R. Zhang, R. Gangi Reddy, M. A. Sultan, V. Castelli, A. Ferritto, R. Florian, E. Sarioglu Kayi, S. Roukos, A. Sil, and T. Ward, "Multi-stage pre-training for low-resource domain adaptation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Online), pp. 5461–5468, Association for Computational Linguistics, Nov. 2020.

[14] W. Tai, H. T. Kung, X. Dong, M. Comiter, and C.-F. Kuo, "exBERT: Extending pre-trained models with domain-specific vocabulary under constrained training resources," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, (Online), pp. 1433–1439, Association for Computational Linguistics, Nov. 2020.

[15] V. Sachidananda, J. Kessler, and Y.-A. Lai, "Efficient domain adaptation of language models via adaptive tokenization," in *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, (Virtual), pp. 155–165, Association for Computational Linguistics, Nov. 2021.

[16] B. Minixhofer, F. Paischer, and N. Rekabsaz, "WECHSEL: Effective initialization of subword embeddings for cross-lingual transfer of monolingual language models," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, (Seattle, United States), pp. 3992–4006, Association for Computational Linguistics, July 2022.

[17] OpenAI, "ChatGPT: Conversational ai language model." `https://www.openai.com/research/chatgpt`, 2023. Accessed on May 31, 2023.

[18] D. Paperno, G. Kruszewski, A. Lazaridou, N. Q. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fernández, "The LAMBADA dataset: Word prediction requiring a broad discourse context," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, (Berlin, Germany), pp. 1525–1534, Association for Computational Linguistics, Aug. 2016.

[19] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

초록

사전 훈련된 언어 모델(PLM)은 시간이 지남에 따라 새로운 데이터에 대한 성능이 낮아질 수 있으므로 새로운 단어나 개념에 대해 PLM을 적응시킬 필요성이 대두되었다. PLM을 새로운 어휘에 적응시키는 효율적인 접근 방식이 도메인 또는 교차 언어 적응 분야에서 연구되었지만, 이러한 방법은 어휘 업데이트가 적시에, 주기적으로, 소규모 (1MB가량의 학습 데이터로 1-10개의 새 단어 추가)로 발생해야 하는 설정에서는 아직 실험된 적이 없다. 위와 같은 방법들은 새로운 세팅에서 만족스럽지 못한 성능을 나타내거나 새로운 어휘에 과적합되는 결과를 낳는 경향을 보였다. 더 나아가, 기존의 PLM의 모델 편집 테크닉 또한 사전 학습 도중에 보지 못한 정보를 PLM에 주입하는 데에는 효과적이지 못한 것으로 알려져있다. 따라서 본 논문에서는, 인간이 새로운 단어나 개념을 접할 때 내부 지식을 처리하는 방식을 모방하여 PLM을 새로운 어휘(즉, 신조어)에 적응시키기 위한 맞춤형 방법 W-SUM 을 제안한다. Piaget의 인지 발달 이론에서 동화에 영감을 받아, W-SUM 는 기존 토큰 임베딩의 가중 합계를 통해 새로운 어휘의 최적 임베딩을 찾고, 이 과정에서 PLM의 기존 토큰 임베딩에 내재된 풍부한 지식을 활용하도록 한다. 가중 합계를 위한 가중치는 PLM이 추가적인 사전학습을 통해서 찾도록 했다. ECBD와 LAMBADA라는 두 가지 언어 모델 조사 태스크에서 W-SUM 를 평가하고 임베딩 비교 분석을 통해 W-SUM 이 새로운 어휘에 대한 좋은 임베딩을 획득하는데 효과적이라는 것을 보인다.

주요어:

학번: 2021-23818