Efficient Domain Adaptation of Language Models via Adaptive Tokenization

Vin Sachidananda[*]
Stanford University
vsachi@stanford.edu

Jason S. Kessler
Amazon
jasokess@amazon.com

Yi-An Lai
AWS AI HLT
yianl@amazon.com

Abstract

Contextual embedding-based language models trained on large data sets, such as BERT and RoBERTa, provide strong performance across a wide range of tasks and are ubiquitous in modern NLP. It has been observed that fine-tuning these models on tasks involving data from domains different from that on which they were pretrained can lead to suboptimal performance. Recent work has explored approaches to adapt pretrained language models to new domains by incorporating additional pretraining using domain-specific corpora and task data. We propose an alternative approach for transferring pretrained language models to new domains by adapting their tokenizers. We show that domain-specific subword sequences can be efficiently determined directly from divergences in the conditional token distributions of the base and domain-specific corpora. In datasets from four disparate domains, we find adaptive tokenization on a pretrained RoBERTa model provides >97% of the performance benefits of domain specific pretraining. Our approach produces smaller models and less training and inference time than other approaches using tokenizer augmentation. While adaptive tokenization incurs a 6% increase in model parameters in our experimentation, due to the introduction of 10k new domain-specific tokens, our approach, using 64 vCPUs, is 72x faster than further pretraining the language model on domain-specific corpora on 8 TPUs.

1 Introduction

Pretrained language models (PLMs) trained on large "base" corpora, oftentimes >100GB of uncompressed text Liu et al. (2019); Brown et al. (2020), are used in many NLP tasks. These models first learn contextual representations in an unsupervised manner by minimizing a masked language modeling objective over a base corpus. This stage of unsupervised language model training is referred to as "pretraining". Subsequently, for supervised classification tasks, the output head of this pretrained model is swapped for a lightweight classifier and trained further on a classification objective over labeled data, referred to as "fine-tuning".

Recent work has examined the transferability of PLMs Gururangan et al. (2020) and their contextual representations to domains differing from their base corpora. On text classification tasks from four different domains, it was shown that continuing to pretrain RoBERTa's contextual embeddings on additional domain (DAPT) and/or task-specific data (TAPT) resulted in performance gains over only fine-tuning a baseline RoBERTa model. These performance gains, however, were inferior to each task's start-of-the-art metrics which were largely based on training versions of RoBERTa, or other LMs, from scratch on a large sample of in-domain data.

These performance gains come at substantial financial, time, and environmental costs in the form of increased computation, with pretraining an LM from scratch being the most expensive, using additional pretraining in the middle, and only fine-turning an off-the-shelf model the most economical.

One observed advantage Gu et al. (2020) that pretraining from scratch on in-domain data has over continual pretraining is that the tokenizer's vocabulary captures domain-specific terms. This allows semantics of those terms to be directly learned in their fixed embeddings, and relieves the language model from having to encode these semantics through the contextual embeddings of these domain-specific term's subwords. Recent work Zhang et al. (2020); Poerner et al. (2020) has shown adding whole words common to the target domain but absent from a PLM's tokenizer improves performance on single tasks. In this work, we show that augmenting an PLM with statistically derived subword tokens selected for domain association with

[*] Work done during an internship at Amazon.

Proceedings of the 2nd Workshop on Simple and Efficient Natural Language Processing, pages 155–165
November 10, 2021. ©201 Association for Computational Linguistics

simple embedding initializations and no further pretraining provide an effective means of adapting a PLM across tasks and domains. In contrast, both Zhang et al. (2020) and Poerner et al. (2020) add inefficiencies by respectively requiring further masked language model (MLM) pretraining and doubling the resources needed for inference.

In this paper, we efficiently adapt a PLM by simply augmenting its vocabulary with domain-specific token sequences. We find that this adaptation, which requires no further pretraining, rivals the accuracy of domain and task-adapted pretraining approaches proposed in Gururangan et al. (2020) but requires only a small fraction of the compute cost.

2 Related work

Gururangan et al. (2020) describes two complementary methods using a task's training data or a separate unlabeled domain-specific corpus to further pretrain an LM, denoted as Task-Adaptive Pretraining (TAPT) and Domain-Adaptive Pretraining (DAPT) respectively. This paper shows the value of employing additional in-domain data in pretraining on four domains relative to only fine-tuning a PLM. Our approach is directly comparable to DAPT, as we only use in-domain corpora for adaptation.

Zhang et al. (2020) augment RoBERTa's vocabulary with in-domain OOV whole words. The most frequently occurring whole words are added until the OOV rate drops to 5% on the task corpus. They randomly initialize weights and pretrain a model. This improves performance on TechQA and AskUbuntu. Tai et al. (2020) also augmented BERT with tokens selected by frequency (12k OOV wordpieces were used) and pretrained a modified version of BERT which allowed for only new token's embeddings to be modified while the original embeddings remained fixed. They found that using more than 12k augmented tokens didn't improve their biomed NER and relation extraction performance, and that, once augmented, performance improved with more pretraining (4-24 hours were studied.)

Poerner et al. (2020) augment BERT's vocabulary with all in-domain OOV whole words, adding 31K tokens to bert-base-cased's 29K wordpieces. They trained a word2vec model on an in-domain corpus and fit a linear transformation to project the word embeddings into the model's input embedding space. No further pretraining is done, but during finetuning, the original tokenizer and the adapted tokenizer are both used. For inference, the finetuned model is run with both the original tokenizer and the adapted tokenizer and the outputs are averaged. Their F1 score outperforms BERT on all eight biomedical NER tasks studied. The approach has the disadvantage of increasing the parameter size of bert-base-cased by 2.2x due to the embeddings of added tokens and doubles the resources needed for inference.

Hofmann et al. (2021) demonstrates how Wordpiece tokenization does not capture the semantics of derivationally complex words as well as an approach using a modified version of Wordpiece designed to produce subword segmentations consisting of linguistic prefixes, suffixes and affixes Hofmann et al. (2020). This subword tokenizer outperformed WordPiece in determining words' polarity or their source domains. Experiments were conducted on novel embedding tokens in BERT via approaches including a projection-based method and mean pooling (both similar to §3.3).

Training language models from scratch in the domain of interest has been shown to provide improved in-domain performance when compared to out-of-domain PLMs Huang et al. (2019). In addition to Gururangan et al. (2020), prior work has shown the effectiveness of continued pretraining for domain adaptation of PLMs Alsentzer et al. (2019); Chakrabarty et al. (2019); Lee et al. (2019). For the task of Aspect-Target Sentiment Classification, Rietzler et al. (2020) uses both DAPT and task-specific fine-tuning in order to adapt language models representations. Identifying domain-characteristic words is a well-studied problem, and many metrics have been proposed for this task through comparing the distributions of tokens in contrasting corpora Rayson et al. (1997); Monroe et al. (2008); Kessler (2017). Muthukrishnan et al. (2008) used the pointwise KL-divergence to distinguish informativeness of key phrase candidates in a domain corpus relative to a background.

3 Adaptive tokenization of contextual embeddings

We define adaptive tokenization (AT) as the process of augmenting a PLM's tokenizer and fixed subword embeddings with new entries taken from a novel corpus. AT consists of two goals which must be achieved for domain adaptation. First, selection of domain-specific tokens, with which to augment a pretrained tokenizer, from an in-domain corpus must be determined. Second, an appropriate initialization in the input space of the contextual embedding models needs to be determined for additions to the tokenizer vocabulary. In this section, we detail approaches for each of these linked tasks.

3.1 Tokenizer vocabulary augmentation

In this section, we detail approaches for identifying domain-specific token sequences to be added during tokenizer augmentation. Common tokenization schemes such as Byte Pair Encoding Sennrich et al. (2016) and WordPiece Schuster and Nakajima (2012); Wu et al. (2016) are greedy algorithms and, as a result, merge subwords into individual tokens if such a sequence occurs with high relative frequency. When adapting a tokenizer our goal is to identify subword sequences which occur with high relative frequency in a domain specific corpus compared to the pretraining corpus. In Table 1, we provide the corpora for each domain in which experimentation is conducted. Next, we show how to operationalize this framework to find domain-specific token sequences.

3.2 Identifying domain-specific token sequences

In this section, we detail our approach for selection of token sequences which are both difficult to represent in a base tokenizer and have large disparities in occurrence between domain-specific and base corpora. Conceptually, we would like to add new tokens to the source tokenizer which are sequences of existing tokens and, in the in-domain corpus, are extensions of existing token sequences.

(I) Computing Empirical Token Sequence Distributions We first compute counts of sequences of $[1, \lambda]$ subword tokens (s) in each corpus C, namely the source corpus for RoBERTa (S) and the in-domain corpus which is the target of our adaptation (D). The source language model's tokenizer (namely Roberta-base) is used as the source of subword tokens. The counts of each subtoken sequences are represented as C_s, where C is the corpus and ss is the subword sequence. If s does not appear in C, $C_s = 0$. We only retain sequences occurring at least $\phi = 20$ times in one corpus. The maximum subword token sequence length (λ) is 10. We limit subtoken sequences to word boundaries as detected through whitespace tokenization.

Next, we predict how "phrase-like" a sequence of tokens C_s is, using a probability $P_C(s)$. Define

$$P_C(s) = \frac{C_s}{C_t}$$

where t is first $|s|-1$ subtoken sequence of s. These probabilities should be thought of as the surprise of the sequence s in the corpus being counted and are indicative of the how phrase-like s is.

As an example, consider a hypothetical corpus consisting of documents written about classical music. Roberta-base's tokenizer splits "oboe" into the subtokens $\langle \text{ob}, \text{oe} \rangle$. In this classical music corpus, the portion of tokens following "ob" which are "oe" (composing in the word "oboe") is surely much higher than in a general base corpus where other words staring with the "ob" subtoken like "obama" (tokenized as $\langle \text{ob}, \text{ama} \rangle$) are much more frequent and "oboe" much less.

(II) Domain shift scoring of Token Sequence Distributions with Conditional KL Divergence In order to characterize these differences in probabilities, we use the pointwise KL-divergence. Letting p and q be probabilities, the pointwise KL-divergence is defined as:

$$D_{KL}(p \parallel q)) = p \log \frac{p}{q}$$

Let the sequence relevance score $R(s)$ be defined as

$$R(s) = D_{KL}(P_D(s) \parallel P_S(s)).$$

$R(s)$ indicates how much the phrase-like probability of sequence s in the in-domain corpus D ($P_D(s)$) diverges from the baseline phrase-like probability of s in the base corpus S.

(III) Selection of Token Sequences for Tokenizer Augmentation For all experiments, we add the $\eta = 10K$ sequences with the largest R, sorted irrespective of sequence length, to the domain-augmented tokenizer.

This introduces of 7.68M parameters (embedding size 768 × 10K new tokens), a 6% increase over Roberta-base's 125M.[1]

3.3 Initialization approaches for AT

In this section, we provide two approaches to impute contextual embedding input representations for tokens added in §3.1.

Subword-based initialization In this common initialization Casanueva et al. (2020); Vulić et al.

[1] github.com/pytorch/fairseq/tree/master/examples/roberta

Algorithm 1 Selection of Domain-Specific Token Sequences for Tokenizer Augmentation

```
Require: Base Tokenizer $Tok$, Base LM $LM_{base}$, Base and Domain Unigram Dists. $U_{base}, U_{domain}$,
  Base and Domain Seq. Dists. $T_{base}$= { }, $T_{domain}$= { } Min. Seq. Frequency $F_{min}$, # Aug. to make $N$,
  Max Aug. Length $L$, Augmentations = []
  (I) Computing Empirical Token Sequence Distributions
  for word, count $(w, count)$ in $U_{base}$ do                ▷ Do the same for Domain Corpus
      $Seq[t_0, t_1, ..., t_n] := Tok(w)$
      for i in [1,n] do
          $T_{base}[Seq[: i]] += count$
      end for
  end for
  $T_{domain}$.values() $/=$ sum($U_{domain}$.values())        ▷ Normalize Sequence Distributions
  $T_{base}$.values() $/=$ sum($U_{base}$.values())
  (II) Domain shift scoring of Token Seq. Dists. with Conditional KL Divergence
  $Score_{DKL} = \{\}$
  for Seq in $T_{base} \bigcap T_{domain}$ do
      $Score_{DKL}[Seq] := T_{domain}[Seq] * \log \frac{T_{domain}[Seq]}{T_{base}[Seq]}$
  end for
  (III) Selection of Token Sequences for Augmentation
  SortDescending($Score_{DKL}$)
  for Seq in $Score_{DKL}$ do
      if Len(Augmentations) $= N$ then
          break
      end if
      if Len($Seq$) $< L$ AND $T_{domain} > F_{min}$ AND $T_{base} > F_{min}$ then
          Augmentations.append(Seq)
      end if
  end for
  return Augmentations
```

(2020); Hofmann et al. (2021), additions to the tokenizer are embedded as the mean of their Roberta-base fixed subword embeddings. In cases where all a novel word's subwords are unrelated to its specific, in-domain meaning, this initialization may cause unwanted model drift in fine-tuning for unrelated tokens with similar fixed embeddings.

Algorithm 2 Projection-Based Initialization of Augmented Tokens

```
Require: LM Input Embeddings $C_s$, Base and
  Domain Learned Input Embeddings $X_s, X_t$,
  and Embedding Size $d$.
  (I) Learn Mapping $\hat{\mathcal{M}}$: $C_s \to X_s$ with SGD:
  $\hat{\mathcal{M}} = \arg\min_{\mathcal{M} \in \mathbb{R}^{d \times d}} \|\mathcal{M}X_s - C_s\|_F$
  (II) Get Inits. for Aug. Tokens using $\hat{\mathcal{M}}$:
  $C_t = \hat{\mathcal{M}} X_t$
  return $C_t$
```

Projection-based initialization To mitigate possible issues with averaging subword embeddings, we also consider projections between static token embeddings to the input space of contextual embeddings, similar to Poerner et al. (2020).

To summarize this approach, our goal is to learn a mapping between the input token embeddings in RoBERTa, C_{base}, and word2vec token embeddings learned independently on the base[2] and domain specific corpora, X_{base}, X_{domain}. The tokens in C_{base} include the original RoBERTa tokens while those in X_{base} and X_{domain} include both the original RoBERTa tokens and the augmented tokens found using adaptive tokenization detailed in §3.2. First, a mapping M, parametrized as a single layer fully connected network, from X_{base} to C_{base} is learned which minimizes distances, on the original set of tokens in RoBERTa. The goal of this mapping is to learn a function which can translate

[2] See §5.4 for how the RoBERTa source corpora is approximated to form our base corpus.

Domain	Pretrain Corpus [# Tokens]	Task	Task Type	Train (Lab.)	Dev.	Test	Classes
BioMed	1.8M papers from S2ORC [5.1B]	ChemProt	relation classification	4169	2427	3469	13
		RCT	abstract sent. roles	18040	30212	30135	5
CS	580K papers from S2ORC [2.1B]	ACL-ARC	citation intent	1688	114	139	6
		SciERC	relation classification	3219	455	974	7
News	11.9M articles [6.7B]	HyperPartisan	partisanship	515	65	65	2
Reviews	24.75M Amazon reviews [2.1B]	IMDB	review sentiment	20000	5000	25000	2

Table 1: Specifications of the various target task and pretraining datasets to replicate experiments in Gururangan et al. (2020). Due to the restrictions on accessible papers in S2ORC, we are using versions of BioMed and CS which are approximately 33% and 74% smaller than were used in Gururangan et al. (2020). Sources: S2ORC Lo et al. (2020), News Zellers et al. (2019), Amazon reviews He and McAuley (2016), CHEMPROT Kringelum et al. (2016), RCT Dernoncourt and Lee (2017), ACL-ARC Jurgens et al. (2018), SCIERC Luan et al. (2018), HYPERPARTISAN Kiesel et al. (2019), and IMDB Maas et al. (2011).

word2vec token embeddings to the input space of RoBERTa. Then, the learned mapping M is applied to X_{domain} in order to obtain initializations in the input space of RoBERTa for the augmented tokens found using the approach in §3.2. The operations involved in this approach are detailed in Algorithm 2.

4 Experimentation

In this section, we perform evaluation of our adaptation approach on six natural language processing tasks in four domains, BioMedical, Computer Science, News, and Reviews, following the evaluations in Gururangan et al. (2020). Due to resource constraints, we perform experimentation on all datasets in Gururangan et al. (2020) excluding the Helpfulness dataset from the reviews domain and the Hyperpartisan dataset in the news domain. Each of the excluded datasets contain greater than 100K training examples, resulting in greater than 12 hours of time required for finetuning on 8 Tesla V100 GPUs for a single seed.

Approaches Roberta-base, a commonly used PLM with high performance, is used as a baseline on which supervised finetuning is performed separately for each dataset. Additionally, we compare AT to the DAPT method from Gururangan et al. (2020). As we do not make use of task specific data (i.e., the training data used in fine-tuning), AT is comparable to DAPT in terms of the data utilized. We focus on using large, in-domain data sets which are commonly used in further pretraining (rather than variably sized task-data) since their size both allows for reliable extraction of characteristic subtoken sequences to use in tokenizer augmentation. Adaptive tokenization for task-specific data is future work.

Classification Architecture We use the same classification architecture as in Gururangan et al. (2020), originally proposed in Devlin et al. (2019), in which the final layer's [CLS] token representation is passed to a task-specific feed forward layer for prediction. All hyperaparameters used in experimentation are equivalent to either the "mini", "small", or "big" hyperparameter sets from Gururangan et al. (2020).

Results We find that adaptive tokenization improves performance when compared to the baseline RoBERTa model in all four of the domains on which experimentation is performed. AT provides 97% of the aggregate relative improvement attained by DAPT respectively over Roberta-base while providing an order of magnitude efficiency gain detailed in Table 3. We do not see a significant difference in the performance of AT models based on the Mean or Proj initialization schemes. Given that Mean initialization required half the time as Proj, we recommend its use over Proj.

5 Discussion

5.1 Resource Efficiency in LM Adaptation

Current approaches for training and adapting LMs have resulted in negative environmental impact and high computational resource budgets for researchers. PLMs incur significant compute time during pretraining, typically requiring numerous days of training on $\geq$ 8 GPUs or TPUs Liu et al. (2019); Devlin et al. (2019); Gururangan et al. (2020). In Table 3, we provide a runtime comparison between continued pretraining and AT. We find that AT provides a 72x speedup compared to DAPT and does not require a GPU or TPU to run. The most resource-intensive portion of this procedure involves indexing the corpora and conducting

Domain	Task	RoBERTa	DAPT	TAPT	DAPT + TAPT	AT (Mean)	AT (Proj)	State-of-the-art (in 2020)
BioMed*	ChemProt	$81.9_{1.0}$	$\underline{84.2_{0.2}}$	$82.6_{0.4}$	$\mathbf{84.4}_{0.4}$	$83.6_{0.4}$	$83.1_{0.3}$	84.6
	RCT	$87.2_{0.1}$	$\underline{87.6_{0.1}}$	$87.7_{0.1}$	$\mathbf{87.8}_{0.1}$	$87.5_{0.4}$	$\underline{87.6_{0.3}}$	92.9
CS*	ACL-ARC	$63.0_{5.8}$	$\underline{75.4_{2.5}}$	$67.4_{1.8}$	$\mathbf{75.6}_{3.8}$	$70.1_{2.0}$	$68.9_{1.6}$	71.0
	SciERC	$77.3_{1.9}$	$80.8_{1.5}$	$79.3_{1.5}$	$81.3_{1.8}$	$\underline{\mathbf{81.4}_{0.4}}$	$81.2_{1.2}$	81.8
News	HyperPartisan	$86.6_{0.9}$	$88.2_{5.9}$	$90.4_{5.2}$	$90.0_{6.6}$	$\underline{\mathbf{93.1}_{4.2}}$	$91.6_{5.5}$	94.8
Reviews	IMDB	$95.0_{0.2}$	$95.4_{0.1}$	$95.5_{0.1}$	$\mathbf{95.6}_{0.1}$	$95.4_{0.1}$	$\underline{95.5_{0.1}}$	96.2

Table 2: Results of different adaptive pretraining methods compared to the baseline RoBERTa. AT with mean subword and projective initializations are denoted as AT (Mean) and AT (Proj) respectively. Stddevs are from 5 seeds. Results for DAPT, TAPT, DAPT+TAPT, and state-of-the-arts are quoted from Gururangan et al. (2020). The highest non-state-of-the-art result is bolded, since the state-of-the-art functions as a performance ceiling, leveraging both domain-specific pretraining and an adapted tokenizer. The best of the three approaches which utilize only source and domain domain data before fine-tuning (i.e., DAPT and AT) is underlined. *Due to restrictions on accessible papers in S2ORC, The BioMed and CS pretraining corpora used were respectively 33% and 74% smaller than the versions in Gururangan et al. (2020). Note that state-of-the-art numbers are current at the time of Gururangan et al. (2020), and are from the following works: ChemProt: S2ORC-BERT Lo et al. (2020), RCT: Sequential Sentence Classification Cohan et al. (2019), ACL-ARC: SciBert Beltagy et al. (2019), SciERC: S2ORC-BERT Lo et al. (2020), HyperPartisan: Longformer Beltagy et al. (2020), IMDB: XLNet Large Yang et al. (2019).

Method	Hardware Specs.	Runtime [h:m:s]
DAPT	8x TPU V-3	94 hours
AT (Mean)	64x vCPUs	1:17:35
AT (Projection)	64x vCPUs	4:54:58

Table 3: Runtime and hardware specifications for AT compared to DAPT. The vast majority of the time is spent reading the corpus and creating token distributions. Runtimes are based on the CS 8.1B token corpus. The DAPT runtime is mentioned in Github Issue 16 in Gururangan et al. (2020) and the AT runtimes are linearly extrapolated (an overestimate) from our observed runtime on the open version of CS, a 2.1B token corpus. We needed to perform this extrapolation since the full CS corpus which was used to benchmark Gururangan et al. (2020) is unavailable in S2ORC. "64x vCPUs" indicate the equivalent of an AWS ml.m5.16xlarge EC2 instance was used to determine which subtoken sequences to use for vocabulary augmentation and compute their embeddings. The times reported for AT (Mean) and AT (Projection) where from a single run, with precomputed base corpus token counts and embeddings.

subtoken sequence counts.

In addition to time and resources, the environmental impact of pretraining BERT with a single set of hyperparameters incurs a carbon footprint of approximately 1.5K pounds of CO_2 emissions, more than the average monthly emissions of an individual Strubell et al. (2019). Continued pretraining, which has a similar resource budget to BERT, exacerbates this problem Schwartz et al. (2019). Lastly, we find that the cloud computing costs associated with continual pretraining for both a single domain and set of hyperparameters are $750 compared to around $4.77 (using a ml.m5.16xlarge EC2 instance for 1:17) for AT on cloud computing platforms when using non-preemptible instances. High costs associated with the training of NLP models has led to inequity in the research community in favor of industry labs with large research budgets Strubell et al. (2019).

5.2 Augmented Token Sequences selected in each domain

In Table 4, we provide examples of augmented vocabulary selected by our adaptive tokenization algorithm for each of the four domains used in experimentation. In each domain, the augmented tokens identified by AT correspond to domain-specific language. For instance, augmented tokens in the Reviews domain token sequences often contain contractions such as "I've" and "it's", which are frequently used in informal language. In the News domain, augmented tokens include financial terms such as "NYSE" and "Nasdaq" along with media outlets such as "Reuters" and "Getty". Many of the augmented tokens in the Computer Science domain are mathematical and computing terms such as "Theorem", "Lemma", "Segmentation", and "Gaussian". Lastly, augmented tokens in the BioMedical domain are largely concerned with biological mechanisms and medical procedures such as "phosphorylation", "assays", and "transfect".

5.3 Future directions

While we have evaluated this approach on Roberta-base, it can be used on any PLM which uses subword tokenization. It would be interesting future

BioMed	CS	News	Reviews
[inc, ub, ated] → incubated	[The, orem] → Theorem	[t, uesday] → tuesday	[it, ’s] → it’s
[trans, fect] → transfect	[L, em, ma] → Lemma	[ob, ama] → obama	[that, ’s] → that’s
[ph, osph, ory] → phosphory	[vert, ices] → vertices	[re, uters] → reuters	[sh, oes] → shoes
[mi, R] → miR	[E, q] → Eq	[iph, one] → iphone	[doesn, ’t] → doesn’t
[st, aining] → staining	[cl, ust, ering] → clustering	[ny, se] → nyse	[didn, ’t] → didn’t
[ap, opt, osis] → apoptosis	[H, ence] → Hence	[get, ty] → getty	[can, ’t] → can’t
[G, FP] → GFP	[Seg, mentation] → Segmentation	[inst, agram] → instagram	[I, ’ve] → I’ve
[pl, asm] → plasm	[class, ifier] → classifier	[bre, xit] → brexit	[b, ought] → bought
[ass, ays] → assays	[Ga, ussian] → Gaussian	[nas, daq] → nasdaq	[you, ’ll] → you’ll
[ph, osph, ory, lation] → phosphorylation	[p, olyn] → polyn	[ce, o] → ceo	[kind, le] → kindle

Table 4: Samples of token sequences with large JSD between base and domain corpora sequence distributions; all of these sequences were added during AT to the Roberta-Base tokenizer.

work to see if the performance gain will hold on larger PLMs with richer vocabularies or on smaller PLMs. One may speculate the benefit of AT is due to encoding non-compositional subword tokens in the input embedding space. And furthermore, this lifts some of the responsibility for encoding their semantics from the LM’s interior weights. Since these non-compositional tokens are characteristic to the domain corpus, their representations may be important to the end task and and need to be learned or improved during fine-tuning. If this is the case, then perhaps models with fewer interior weights benefit more from AT since the connection between the non-compositional tokens would be built into the input, allowing interior weights to better learn the semantics of novel non-compositional tokens and opposed to also having to learn the component tokens’ connection.

While this work tests AT on an English language PLM, it can hypothetically be applied to any PLM regardless of its source language(s). Exploring how AT can work with additional pretraining on domain data is clear future work. Tai et al. (2020) show that specialized further pretraining on domain data on using a model augmented with domain characteristic whole word tokens results in an improved performance/pretraining time curve. It would also be fruitful to explore how that curve changes when using more efficient pretraining techniques such as in Clark et al. (2020).

While we compared different novel token sequence embedding techniques, we did not study different ways of identifying subtoken sequences to add. Comparing AT to approaches such adding whole word tokens Tai et al. (2020) would confirm our hypothesis that phrase-like token sequences are useful.

Experimenting with the number of subtoken sequences added to the tokenizer (η fixed at $10K$) may also be worthwhile. While Tai et al. (2020) found $12K$ tokens additions optimal, Poerner et al. (2020) added $310K$ tokens. Seeing the trade-off between added tokens and performance would be useful, as each additional parameter increases the model size.

Our approach requires new tokens to appear ϕ times in both the source and domain corpora. While this was necessary in order to produce source-corpus word embeddings in Proj, it does not allow for domain-exclusive subtoken sequences to be added to the tokenizer. Abandoning this requirement for Mean may lead to a better set of token augmentations.

We can also experiment with other subtoken candidate selection techniques. For example, Schwartz et al. (2013) used pointwise mutual information (PMI) to determine how phrase-like candidates word sequences were. PMI is the log ratio of the probability of a phrase vs. the product of the probability of its component unigrams. While our approach considers the probability of a subtoken given a preceding sequence, it, unlike PMI, does not consider the probability of that following subtoken in isolation. This may lead to domain-specific subtokens sneaking into augmented token sequences, such as the contraction tokens added to the reviews Reviews tokenizer in Table 4.

5.4 Implementation details

The code is in preparation for release. The hyperparameter search used was ROBERTA_CLASSIFIER_MINI from Gururangan et al. (2020) from their codebase `https://github.com/allenai/dont-stop-pretraining`. Token counts for RoBERTa-base were estimated using English Wikipedia 20200501.en and an open source book corpus from `https://storage.googleapis.com/huggingface-nlp/datasets/bookcorpus/bookcorpus.tar.bz2`. Word2vec embeddings were computed with Gensim (Rehurek and Sojka, 2011), using the following parameters: `Word2Vec(..., size=768, window=5, min_count=100, epochs=2, sample=1e-5)`

6 Conclusion

In this paper, we introduced adaptive tokenization (AT) a method for efficiently adapting pretrained language models utilizing subword tokenization to new domains. AT augments a PLM's tokenization vocabulary to include domain-specific token sequences. We provide two approaches for initializing augmented tokens: mean subword and projections from static subword embeddings. AT requires no further language model pretraining on domain-specific corpora, resulting in a 38x speedup over pretraining on the corpora without specialized hardware. Across four domains, AT provides >97% of the performance improvement of further pretraining on domain-specific data over Roberta-base. This initial work suggests that adapting the subword tokenization scheme of PLMs is an effective means of transferring models to new domains. Future work entails hybrid approaches using both AT and small amounts of LM pretraining, alternative metrics for augmented token selection, improved initialization of augmented token representations, and the use of task data.

Acknowledgements

We thank Yi Zhang, William Headden, Max Harper, Chandni Singh, Anuj Ahluwalia, Sushant Sagar, Jay Patel, Sachin Hulyalkar, and the anonymous reviewers for their valuable feedback.

Ethics statement

As mentioned in §5, pretrained language models incur significant costs with respect to time, computational resources and environmental impact. Continued domain specific pretraining, which has a similar resource budget to BERT, exacerbates this problem Schwartz et al. (2019). In this work, we provide approaches for adapting pretrained language models to new domains with an approach, Adaptive Tokenization, which seeks to minimize costs associated with continued domain specific pretraining. It should be noted that we do not decrease the resource and environmental associated with pretraining, only the costs for domain adaptive pretraining which are nevertheless sizable (e.g. 32 TPU days for DAPT).

Additionally, we find that the cloud computing costs associated with continued domain specific pretraining on a single domain and set of hyperparameters are around $750 compared to around $5 for AT on a cloud computing platform. High costs associated with the training of NLP models has led to inequity in the research community in favor of industry labs with large research budgets Strubell et al. (2019), a problem we seek to ameliorate.

This work does not address the high resource cost in fine-tuning PLMs. Risks associated with this paper are that this work may encourage the use of PLMs in more settings, such as domains with small amounts of data, and introduce potentially harmful inductive biases which have been found in many commonly used PLMs.

We include statistics about the data sets used in Table 1, these data sets were introduced in Gururangan et al. (2020) and open source.

References

Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. 2019. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A pretrained language model for scientific text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3615–3620, Hong Kong, China. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Iñigo Casanueva, Tadas Temčinas, Daniela Gerz, Matthew Henderson, and Ivan Vulić. 2020. Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Tuhin Chakrabarty, Christopher Hidey, and Kathy McKeown. 2019. IMHO fine-tuning improves claim detection. 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 558–563, Minneapolis, Minnesota. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Arman Cohan, Iz Beltagy, Daniel King, Bhavana Dalvi, and Dan Weld. 2019. Pretrained language models for sequential sentence classification. In *EMNLP*.

Franck Dernoncourt and Ji Young Lee. 2017. PubMed 200k RCT: a dataset for sequential sentence classification in medical abstracts. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 308–313, Taipei, Taiwan. Asian Federation of Natural Language Processing.

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.

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Valentin Hofmann, Janet Pierrehumbert, and Hinrich Schütze. 2020. DagoBERT: Generating derivational morphology with a pretrained language model. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3848–3861, Online. Association for Computational Linguistics.

Valentin Hofmann, Janet B. Pierrehumbert, and Hinrich Schütze. 2021. Superbizarre is not superb: Improving bert's interpretations of complex words with derivational morphology.

Kexin Huang, Jaan Altosaar, and Rajesh Ranganath. 2019. Clinicalbert: Modeling clinical notes and predicting hospital readmission. *arXiv:1904.05342*.

David Jurgens, Srijan Kumar, Raine Hoover, Dan McFarland, and Dan Jurafsky. 2018. Measuring the evolution of a scientific field through citation frames. *Transactions of the Association for Computational Linguistics*, 6:391–406.

Jason Kessler. 2017. Scattertext: a browser-based tool for visualizing how corpora differ. In *Proceedings of ACL 2017, System Demonstrations*, pages 85–90, Vancouver, Canada. Association for Computational Linguistics.

Johannes Kiesel, Maria Mestre, Rishabh Shukla, Emmanuel Vincent, Payam Adineh, David Corney, Benno Stein, and Martin Potthast. 2019. SemEval-2019 task 4: Hyperpartisan news detection. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 829–839, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Jens Kringelum, Sonny Kim Kjaerulff, Søren Brunak, Ole Lund, Tudor I. Oprea, and Olivier Taboureau. 2016. ChemProt-3.0: a global chemical biology diseases mapping. *Database*, 2016. Bav123.

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. *Bioinformatics*, 36(4):1234–1240.

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. *CoRR*, abs/1907.11692.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel Weld. 2020. S2ORC: The semantic scholar open research corpus. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4969–4983, Online. Association for Computational Linguistics.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3219–3232, Brussels, Belgium. Association for Computational Linguistics.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

B. L. Monroe, Michael Colaresi, and K. Quinn. 2008. Fightin' words: Lexical feature selection and evaluation for identifying the content of political conflict. *Political Analysis*, 16:372–403.

Pradeep Muthukrishnan, Joshua Gerrish, and Dragomir R. Radev. 2008. Detecting multiple facets of an event using graph-based unsupervised methods. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 609–616, Manchester, UK. Coling 2008 Organizing Committee.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. Inexpensive domain adaptation of pretrained language models: Case studies on biomedical NER and covid-19 QA. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1482–1490, Online. Association for Computational Linguistics.

Paul Rayson, G. Leech, and Mary Hodges. 1997. Social differentiation in the use of english vocabulary: some analyses of the conversational component of the british national corpus. *International Journal of Corpus Linguistics*, 2:133–152.

Radim Rehurek and Petr Sojka. 2011. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2).

Alexander Rietzler, Sebastian Stabinger, Paul Opitz, and Stefan Engl. 2020. Adapt or get left behind: Domain adaptation through BERT language model finetuning for aspect-target sentiment classification. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4933–4941, Marseille, France. European Language Resources Association.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5149–5152.

H. Andrew Schwartz, Johannes C. Eichstaedt, Margaret L. Kern, Lukasz Dziurzynski, Stephanie M. Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin E. P. Seligman, and Lyle H. Ungar. 2013. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PLOS ONE*, 8(9):1–16.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2019. Green AI. *CoRR*, abs/1907.10597.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Wen Tai, H. T. Kung, Xin Dong, Marcus Comiter, and Chang-Fu Kuo. 2020. exBERT: Extending pre-trained models with domain-specific vocabulary under constrained training resources. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1433–1439, Online. Association for Computational Linguistics.

Ivan Vulić, E. Ponti, Robert Litschko, Goran Glavas, and Anna Korhonen. 2020. Probing pretrained language models for lexical semantics. *ArXiv*, abs/2010.05731.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *CoRR*, abs/1905.12616.

Rong Zhang, Revanth Gangi Reddy, Md Arafat Sultan, Vittorio Castelli, Anthony Ferritto, Radu Florian, Efsun Sarioglu Kayi, Salim Roukos, Avi Sil, and Todd Ward. 2020. Multi-stage pre-training for low-resource domain adaptation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5461–5468, Online. Association for Computational Linguistics.