TADPOLE: Task ADapted Pre-training via anOmaLy dEtection

Vivek Madan
AWS AI Labs
vivmadan@amazon.com

Ashish Khetan
AWS AI Labs
khetan@amazon.com

Zohar Karnin
AWS AI Labs
zkarnin@amazon.com

Abstract

The paradigm of pre-training followed by fine-tuning has become a standard procedure for NLP tasks, with a known problem of domain shift between the pre-training and downstream corpus. Previous works have tried to mitigate this problem with additional pre-training, either on the downstream corpus itself when it is large enough, or on a manually curated unlabeled corpus of a similar domain. In this paper, we address the problem for the case when the downstream corpus is too small for additional pre-training. We propose TADPOLE, a task adapted pre-training framework based on data selection techniques adapted from *Domain Adaptation*. We formulate the data selection as an anomaly detection problem that unlike existing methods works well when the downstream corpus is limited in size. It results in a scalable and efficient unsupervised technique that eliminates the need for any manual data curation. We evaluate our framework on eight tasks across four different domains: Biomedical, Computer Science, News, and Movie reviews, and compare its performance against competitive baseline techniques from the area of Domain Adaptation. Our framework outperforms all the baseline methods. On small datasets with less than 5K training examples, we get a gain of 1.82% in performance with additional pre-training for only 5% steps. It also compliments some of the other techniques such as data augmentation known for boosting performance when downstream corpus is small; highest performance is achieved when data augmentation is combined with task adapted pre-training.

1 Introduction

Pre-trained language models such as ELMo (Peters et al., 2018), GPT (Radford et al., 2018), BERT (Devlin et al., 2018), Transformer-xl (Dai et al., 2019) and XLNet (Yang et al., 2019) have become a key component in solving virtually all natural language tasks. These models are pre-trained on large amount of cross-domain data ranging from Wikipedia to Book corpus to news articles to learn powerful representations. A generic approach for using these models consists of two steps: (a) Pre-training: train the model on an extremely large general domain corpus, e.g. with masked language model loss; (b) Finetuning: finetune the model on labeled task dataset for the downstream task.

Even though the approach of pre-training followed by fine-tuning has been very successful, it suffers from *domain shift* when applied to tasks containing text from a domain that is not sufficiently represented in the pre-training corpus. An immediate way of solving the problem is to pre-train the model on task domain data instead of the general domain data. For a handful of very popular task domains, the research community invested time and resources to collect a large domain-specific data corpus and pre-train a language model on it. The models include BioBERT pre-trained on biomedical text (Lee et al., 2020), ClinicalBERT pre-trained on clinical notes (Huang et al., 2019), SciBERT pre-trained on semantic scholar corpus (Beltagy et al., 2019), and FinBERT pre-trained on financial documents (Araci, 2019). These models achieve significant gain in performance over a model trained on general domain data, when the downstream task belongs to the respective domains.

These papers demonstrate how useful it can be to shift the domain of the pre-trained model. However, the approach is expensive and time consuming as it requires collecting gigabytes of domain data for each new task. The long-tail of domains remains left behind without a realistic solution. To mitigate this, in absence of the huge task domain data, a different known approach is to collect a medium (MBs, not GBs) amount of unlabeled task data, and adapt the pre-trained (on general data) model by e.g. extending the pre-training procedure on the unlabeled data (Howard and Ruder, 2018; Gururangan et al., 2020). Such task adapted pre-

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 5732–5746
November 7–11, 2021.

training approach achieves relatively smaller gain but is less expensive. Although this approach is cheaper in terms of manual labor when compared to domain adapted BERT, it still requires an effort to collect unlabeled data. It requires much more data than what is needed for only fine-tuning. This is often impossible to achieve, for example when data is highly sensitive. In this paper we propose a solution to this challenging problem, providing domain adaptation to the pre-trained model without the need for any manual effort of data collection.

The high level idea is quite intuitive. Given a generic pre-training data containing text from multiple domains, we filter the available general domain to contain only pieces that are similar to the downstream task corpus. By continuing the pre-training process on this adapted corpus we achieve a better tuned pre-trained model. Figure 1 illustrate the feasibility of this approach with an example downstream task from a medical domain and highlighted text from a news article available in a general domain corpus. The key for a successful implementation is finding the best way of evaluating the similarity of a given snippet to the downstream task.

RCT20K TASK DATA

To investigate the efficacy of 6 weeks of daily low-dose oral prednisolone in improving pain, mobility, and systemic los-grade . . . [OBJECTIVE]
A total of 125 patients with primary knee OA were randomize . . . [METHODS]
Outcome measures included pain reduction and systemic inflammation markers . . . [METHODS]

News Article

For as much as we workout warriors recite that whole "no pain, no gain" mantra, we sure do pop a lot of painkillers. A recent article published in. . . These popular medicines, known as nonsteroidal anti-inflammatory drugs, or NSAIDs, work by suppressing inflammation. . . the article kind of blows past is the fact plenty of racers . . .

Figure 1: Identification of task-data (top panel, medical data) in general domain corpus (bottom panel).

Although not many methods exist to solve the problem of domain shift in the context of pre-training, literature on Domain Adaptation provides several methods for the core task of evaluating the above mentioned similarity. These previous approaches use either a simple language model (LM) (Moore and Lewis, 2010; Axelrod et al., 2011; Duh et al., 2013; Wang et al., 2017b; van der Wees et al., 2017), or a hand crafted similarity score (Wang et al., 2017a; Plank and Van Noord, 2011; Remus, 2012; Van Asch and Daelemans, 2010). The LM based technique are often both over simplistic, and require a fairly large corpus of task data to create a reasonable LM. The hand crafted similarity scores can be seen as ad-hoc methods for distinguishing inliers from outliers (i.e., anomaly detection); they tend to be focused on individual tasks and do not generalize well.

We formulate the similarity evaluation task as that of anomaly detection and propose a Task ADapted Pre-training via anOmaLy dEtection (TADPOLE) framework. Indeed, anomaly detection methods given a domain of instance are able to provide a score for new instances assessing how likely they are to belong to the input domain. We exploit pre-trained models to get sentence representations that are in turn used to train an anomaly detection model. By using pre-trained models, our method is effective even for small text corpora. By taking advantage of existing anomaly detection methods, we replace hand-crafted rules with techniques proven to generalize well. Our approach does not require any manual data curation. To train the anomaly detection method, we only use the task data which is already available as it is necessary for fine-tuning.

In what follows we discuss how we implement our technique and compare it with other data selection methods based on extensive experimental results. We start by filtering out the subset of general domain corpus most relevant to the task. To do this, we explore several anomaly detection methods and give a quantitative criterion to identify the best method for a given task data. Then, we start with a pre-trained model on the general domain corpus and run additional pre-training for only 5% more steps on the filtered corpus from different methods. This is followed by the regular finetuning on the labeled task data. We measure the performance gain as an improvement in accuracy of finetuned model with additional pre-training vs the accuracy of finetuned model without additional pre-training. To establish the performance gain of TADPOLE, we evaluate it on eight tasks across four domains: Biomedical, Computer Science, News, and Movie reviews. We investigate all aspects of TADPOLE by comparing its performance with various baselines based on its variants and the competitive methods available in literature. The main highlights of our work are as follows:

- We provide TADPOLE, a novel anomaly de-

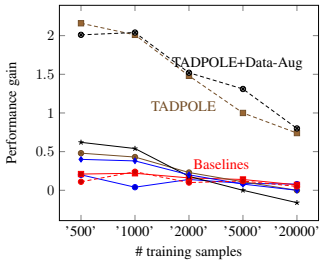

Figure 2: Average performance gain as function of #training samples available for a task.

tection based framework for adapting pre-training for the downstream task. The framework is explained in detail and all its steps are justified via extensive ablation studies.

- TADPOLE is superior to all the baseline methods including (i) LM based relevance score (iii) Distance based relevance score (iii) Continued pre-training on the task data (iv) Data Augmentation while finetuning.
- On tasks with small labeled dataset (less than 5K examples), our method achieves an average 1.82% lift in performance whereas the baselines achieve no more than 0.48%.
- For tasks with large labeled dataset, our method does not depreciate in performance and achieves an average gain of 0.32%. In addition, if only a subset of training samples are available, we observe significantly higher gain. In particular, if only 1000 training samples are available, we observe an average 2.01% gain in performance whereas the baselines achieve no more than 0.24% gain. In all individual tasks, our method is either on par or (statistical) significantly better than all alternatives.
- TADPOLE complements some of the other techniques known for improving performance for small datasets. For instance, TADPOLE performs better than data augmentation and performs even better when combined with data augmentation; Data-Aug $\leq$ TADPOLE $\leq$ TADPOLE + Data-Aug.
- For a task requiring little domain adaptation, GLUE sentiment analysis, our method achieves an improvement of 0.4% in accuracy.

2 Related Work

Since our focus is on Data Selection Methods, we only discuss the related work on Data Selection in Domain Adaptation here. We discuss the other Domain Adaptation techniques in Appendix A.

Data Selection: As discussed above, core of data selection is to determine the relevance weights that in turn modify the source domain to become more similar to the target domain. There has been a sequence of works in trying to find the relevance weights via language models (Moore and Lewis, 2010; Wang et al., 2017b; van der Wees et al., 2017). For instance, Moore and Lewis (2010), Axelrod et al. (2011) and Duh et al. (2013) train two language models, an in-domain language model on the target domain dataset (same as task domain in our case) and an out-of-domain language model on (a subset of) general domain corpus. Then, relevance score is defined as the difference in the cross-entropy w.r.t. two language models. These methods achieve some gain but have a major drawback. A crucial assumption they rely on: *there is enough in-domain data to train a reasonable in-domain language model.* This assumption is not true in most cases. For most tasks, we only have access to a few thousands or in some cases a few hundreds of examples which is not enough to train a reasonably accurate language model. Our techniques rely on text representations based on the available pre-trained model. As such, our similarity score does not rely on models that can be trained with a small amount of data.

Another line of work defines hand crafted domain similarity measures to assign relevance score and filter out text from a general domain corpus (Wang et al., 2017a; Plank and Van Noord, 2011; Remus, 2012; Van Asch and Daelemans, 2010; Gururangan et al., 2020). For instance, Wang et al. (2017a) define the domain similarity of a sentence as the difference between Euclidean distance of the sentence embedding from the mean of in-domain sentence embeddings and the mean of out-of-domain sentence embeddings. Plank and Van Noord (2011) and Remus (2012) define the similarity measure as the Kullback-Leibler (KL) divergence between the relative frequencies of words, character tetra-grams, and topic models. Van Asch and Daelemans (2010) define domain similarity as Rényi divergence between the relevant token frequencies. These are adhoc measures suitable only for the respective tasks, and can be seen as a manual task-optimized anomaly detection. They fail to generalize well for new tasks and domains. Ruder and Plank (2017) attempts to remedy this issue and tries to learn the correct combination of these metrics for each task. They learn the combination

weight vector via Bayesian optimization. However, Bayesian optimization is infeasible for deep networks like BERT. Each optimization step of this process amounts to pre-training the model and fine-tuning it for the task. For Bayesian optimization to work well it requires repeating this process multiple times, which is prohibitively computationally expensive. Thus, they use models such as linear SVM classifier and LDA which do not yield state-of-the-art performance. In contrast, we propose a lightweight method - based on anomaly detection - that can be applied to state-of-the-art deep language models like BERT.

3 TADPOLE: Task ADapted Pre-training via anOmaLy dEtection

Language model and Downstream Tasks. A generic approach for using state-of-the-art language models such as ELMo, GPT, BERT, and XLNet is to pre-train them on an extremely large general domain corpus and then finetune the pre-trained model on the downstream labeled task data. There is evident correlation between model's pre-training loss and its performance on the downstream task after finetuning (Devlin et al., 2018). Our design is motivated by an observation, backed by empirical evidence, that the correlation is even stronger if we consider the pre-training loss not on the pre-training data but the downstream task data.

To make this distinction formal, let $\mathcal{D}$, $\mathcal{D}_{\text{in}}$ be the pre-training and task data. Let Θ denote the parameters of the language model and ℓ_{LM} denote the language model loss function. The pre-training loss on pre-training data ($L_{\text{LM}}(\Theta)$) and target data ($L^{\text{in}}_{\text{LM}}(\Theta)$) are defined as follows: $L_{\text{LM}}(\Theta) = \sum_{x\in\mathcal{D}} \ell_{\text{LM}}(x;\Theta), L^{\text{in}}_{\text{LM}}(\Theta) = \sum_{x\in\mathcal{D}_{\text{in}}} \ell_{\text{LM}}(x;\Theta)$. To show that $L^{\text{in}}_{\text{LM}}(\Theta)$ is better correlated with the performance of the downstream task we consider several BERT language models pre-trained on random combinations of datasets from different domains mentioned in Section 4. Among these we select the BERT models $M_1, \ldots, M_k$ such that selected models have similar Masked Language Model (MLM) loss on the general domain corpus; $|\ell_{LM}(\Theta_i) - \ell_{LM}(\Theta_j)| <= 0.02$. MLM loss of these models on the text from task domain ($\ell^{in}_{LM}(\Theta_i)$) is different. For each M_i, we contrast it with the accuracy/f1 score of the finetuned model on the task data.

We observe in Figure 3 that if we have two equally good language models (similar language model loss on general domain), the language model which is better (tailored) for the task domain has significantly better downstream performance. We

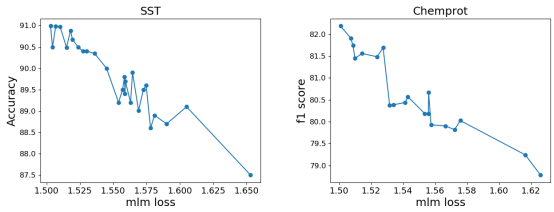

Figure 3: Masked Language Model (MLM) loss of pre-trained BERT on the *task data* vs accuracy/f1 score of corresponding finetuned BERT. Different points correspond to different BERT models, pre-trained on random combination of different datasets. MLM loss on the general domain corpus for all the pre-trained BERT models considered here is roughly the same (within 0.02 of each other).

conclude that in order to ensure success in the downstream task, we should aim to minimize $L^{\text{in}}_{\text{LM}}(\Theta)$. A first attempt would be to pre-train or finetune the language model on $\mathcal{D}_{\text{in}}$. However, training a language model such as ELMo, GPT, BERT or XLNet requires a large corpus with several GBs of text and the available domain specific corpus $\mathcal{D}_{\text{in}}$ is often just the task data which has few MBs of text. Training on such a small dataset would introduce high variance. We reduce this variance by taking training examples from the general domain corpus $\mathcal{D}$, but control the bias this incurs by considering only elements having high relevance to the domain $\mathcal{D}_{\text{in}}$. Formally, we optimize a weighted pre-training loss function

$$L^{\Lambda}_{\text{LM}}(\Theta) = \sum_{x\in D} \lambda(x, \mathcal{D}_{\text{in}}) \cdot \ell_{\text{LM}}(x;\Theta), \qquad (1)$$

where $\lambda(x, \mathcal{D}_{\text{in}})$ are relevance weights of instance x for domain $\mathcal{D}_{\text{in}}$. $\lambda(x, \mathcal{D}_{\text{in}})$ is (close to) 1 if x is relevant to $\mathcal{D}_{\text{in}}$ and (close to) 0 otherwise. We compute these weights using an anomaly detection model fitted on $\mathcal{D}_{\text{in}}$. Note that concept of weighted loss to handle noisy or irrelevant data is well known (Moore and Lewis, 2010; Wang et al., 2017a). Major contribution of our paper is in proposing an anomaly detection based robust approach to find the relevance weights.

3.1 Anomaly Detection to solve the Domain Membership Problem

Detecting whether an instance x is an in-domain instance is equivalent to solving the following problem: *Given task data $\mathcal{T}$ and a sentence s, determine if s is likely to come from the distribution generating $\mathcal{T}$ or if s is an anomaly.*

This view helps us make use of a wide variety of anomaly detection techniques developed in literature (Noble and Cook, 2003; Chandola et al., 2009; Chalapathy and Chawla, 2019). To make use of these techniques, we first need a good numeric representation (embedding) with domain discrimination property. We use pre-trained BERT to embed each sentence into a 768 dimensional vector. Once the data is embedded, we need to decide which among the many anomaly detection algorithms proposed in literature should be applied on the embeddings. To decide the anomaly detection method, we propose an evaluation method ranking the techniques based on their discriminative properties.

Ranking anomaly detection algorithms: The idea is to treat the anomaly score as the prediction of a classifier distinguishing between in-domain and out-of-domain data. By doing so, we can consider classification metrics such as the f1_score as the score used to rank the anomaly detection algorithm. To do this, we split the in-domain data (the task data) into $\mathcal{D}_{\text{in}}^{\text{train}}$, $\mathcal{D}_{\text{in}}^{\text{test}}$ using a 90/10 split. We also create out-of-domain data $\mathcal{D}_{\text{out}}$ as a random subset of $\mathcal{D}$ of the same size as $\mathcal{D}_{\text{in}}^{\text{test}}$. We train an anomaly detection algorithm A with $\mathcal{D}_{\text{in}}^{\text{train}}$, and evaluate it's f1_score on the labeled test set composed of the union $\mathcal{D}_{\text{in}}^{\text{test}} \cup \mathcal{D}_{\text{out}}$, where the labels indicate which set the instance originated from. Note that anomaly detection algorithms considered do not require labeled samples for training. Thus, mixing data from $\mathcal{D}_{out}$ does not add much value.

Table 1 provides the results of this evaluation on six anomaly detection algorithms. Details of the tasks can be found in Section 4. We can see that Isolation Forest consistently performs well for most of the tasks. Local Outlier Factor performs almost equally well but is slower in prediction. Although it is possible to adaptively choose for every task the anomaly detection algorithm maximizing the f1_score, we chose to use a single algorithm, Isolation Forests, for the sake of having a simpler technique and generalizable results. Due to space constraints, we push the discussion on Isolation Forest to Appendix C.

Now that we chose the anomaly detection technique, we move to discuss the effectiveness of the algorithm in (i) identifying the domain from the task data (ii) identifying the domain related data from the general domain corpus.

Figure 4 (left) shows that the anomaly detection algorithm is able to distinguish between the in-task-domain data and out-of-task domain data. These experiments are done for the Sentiment Analysis task (SST) discussed in Section 4. Interestingly, we noticed in our experiments that a language model pre-trained on a diverse corpus is a better choice when compared to a model finetuned on the target domain. We conjecture that the reason is that a finetuned BERT is overly focused on the variations in the task data which are useful for task prediction and forgets information pertaining to different domains which is useful for domain discrimination. We exhibit this phenomenon more clearly in Figure 4 (right) where it is evident that the discriminating ability of the finetuned model is worse.

In order to assess the ability of our model to identify related text we perform the following experiment. First, we create a diverse corpus by taking the union of 4 datasets: News, Finance, CS abstracts and Biology abstracts. Figure 5, column

Task	RC	kNN	PCA	OCS	LOF	IF
CHEMPROT	0.89	0.85	0.92	0.87	0.92	**0.96**
ACL-ARC	0.77	0.88	0.90	0.89	**0.91**	0.88
HYPERPARTISAN	0.86	0.86	0.95	0.98	0.91	**0.98**
RCT20K	0.85	0.88	0.82	0.76	0.87	**0.93**
IMDB	0.88	0.96	0.87	0.81	**0.96**	0.94
SCIERC	0.78	0.84	0.86	0.76	0.88	**0.92**
HELPFULNESS	0.82	0.89	0.83	0.76	0.83	**0.92**
IMDB	0.84	0.89	0.80	0.73	**0.92**	0.87

Table 1: Scores of different anomaly detection algorithms for different tasks. RC: Robust Covariance (Nguyen and Welsch, 2010), kNN: Nearest neighbor (Gu et al., 2019), PCA: Principal Component Analysis (Harrou et al., 2015), OCS: One Class SVM (Schölkopf et al., 2000), LOF: Local Outlier Factor (Breunig et al., 2000), IF: Isolation Forest (Liu et al., 2008)

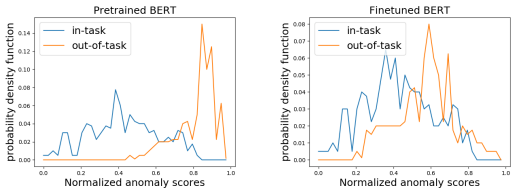

Figure 4: Sentence anomaly scores for SST with anomaly detection algorithm trained on embeddings from Left: pre-trained BERT, Right: finetuned BERT. In-task: sentences from the task data, out-of-task: sentences from general domain corpus.

Corpus	Input data	Filtered (Bio)	Filtered (CS)
News	7.1G (21.8%)	0.1G (2.0%)	0.0G (0.7%)
Finance	4.9G (15.0%)	0.0G (0.1%)	0.0G (0.3%)
CS	8.0G (24.5%)	1.0G (15.4%)	5.1G (78.0%)
Bio	12.6G (38.7%)	5.3G (82.4%)	1.4G (20.9%)

Figure 5: Filtering algorithm trained with Bio Abstracts and CS task data. We mix four corpora, filter out 80% of the data and retain the remaining 20% in both cases.

'Input data' contains their respective sizes. We then train two anomaly score based discriminators, one on CS task data and the other on Bio abstracts. For each model we choose a threshold that would filter out 80% of the data, and observe the data eventually retained by it. The fraction of data retained from each corpus for each model is given in Figure 5, columns 'Filtered (Bio)' and 'Filtered (CS)'. We see that data from the News and Finance corpus is almost completely filtered as it is quite different than the text in abstracts of academic papers. We also see that a non-negligible percent of the filtered data for the Bio model comes from CS and vice versa. Since both corpora are abstracts of academic papers it makes sense that each corpus contains relevant data for the other. The details related to these corpora are given in Appendix B.

3.2 From Anomaly Detection Scores to Domain Adapted Pre-training

Once the anomaly detection object is trained, we use it to compute the relevance weights i.e. compute λ values defined in equation 1 . Let the sentences in the pre-training corpus be $s_1, \ldots, s_N$ with anomaly scores $\{A(s_1), \ldots, A(s_N)\}$. We explore two different strategies of λ value computation. First is when we normalize and transform the scores to compute continuous values and second when we use threshold and compute $0/1$ values.

Continuous λ values: We start by normalizing the anomaly scores to be mean zero and variance 1. Let $\mu = (\sum_{i=1}^{N} A(s_i))/N, \sigma = \sqrt{(\sum_{i=1}^{N}(A(s_i) - \mu)^2)/N}$. Then, for every $i \in \{1, \ldots, N\}$, normalized score is $\bar{A}(s_i) = (A(s_i) - \mu)/\sigma$. Using these normalized sentence anomaly scores, we compute the relevance weights as follows: $\lambda(s_i) = \frac{1}{1+e^{-C(\alpha - \bar{A}(s_i))}}$ where C and α are hyper-parameters. C controls the sensitivity of the weight in terms of anomaly score and α controls the fraction of target domain data present in the general domain corpus. $C \to \infty$ results in $0/1$ weights corresponding to discrete λ setting whereas $C = 0$ results in no task adaptation setting.

Discrete λ values: We sort the sentences as per anomaly scores, $A(s_{\sigma(1)}) \leq A(s_{\sigma(2)}) \leq \cdots \leq A(s_{\sigma(N)})$ and pick β fraction of the sentences with lowest anomaly scores, $\lambda(s_{\sigma(i)}) = 1$ for $i \in \{1, \ldots, \beta N\}$ and 0 otherwise. Even though this approach is less general than the continuous λ values case, it has an advantage of being model independent. We can filter out text, save it and use it to train any language model in a black box fashion. It does not require any change in pre-training or fine-tuning procedure. However, to utilize this option we need to make a change. Instead of filtering out sentences we need to filter out segments containing several consecutive sentences.

To understand why, suppose we filter out sentence 1 and sentence 10 and none of the sentences in between. When we save the text and construct input instances from it for a language model, then an input instance may contain the end of sentence 1 and the start of sentence 10. This is problematic as sentence 1 and sentence 10 were not adjacent to each other in the original corpus and hence, language model does not apply to them. It distorts the training procedure resulting in worse language models. To resolve this issue, we group sentences into segments and classify the relevance of each segment. Formally, let γ be a hyper-parameter and for all $j \in 1, \ldots, \lfloor N/\gamma \rfloor$ let the segment score be $y_j = \sum_{i=(j-1)*\gamma+1}^{j*\gamma} \frac{A(s_i)}{\gamma}$. We sort the segments according to their anomaly scores, $y_{\sigma'(1)} \leq \cdots \leq y_{\sigma'(N/\gamma)}$ and select the β fraction with lowest anomaly scores; save the sentences corresponding to these segments. To completely avoid the issue, we may set segment length very large. However, this is not feasible as the diverse nature of pre-training corpus makes sure that large enough segments rarely belong to a specific domain, meaning that the extracted data will no longer represent our target domain. We experimented with a handful of options for the segment length, and found the results to be stable with segments of 15 sentences.

Continued pre-training instead of pre-training from scratch: Once we have computed the relevance weights $\lambda(s_i)$, we do not start pre-training the language model from scratch as this is not feasible for each new task/domain. Instead, we start with a language model pre-trained on the general domain corpus and perform additional pre-training for relatively fewer steps with the weighted loss function. In our case, we start with a BERT language model pre-trained for one million steps and continued pre-training with updated loss function for either $50,000$ or $100,000$ steps.

4 Experiments

We use datasets listed in Table 2 along with a general domain corpus consisting of 8GB of text from Wikipedia articles. We use $\text{BERT}_{\text{BASE}}$ model provided in the GluonNLP library for all our exper-

Task	Train	Dev	Test	C
HYPERPARTISAN	516	64	65	2
ACL-ARC	1688	114	139	6
SCIERC	3219	455	974	7
CHEMPROT	4169	2427	3469	13
IMDB	20000	5000	25000	2
SST	67349	872	1821	2
AGNEWS	115000	5000	7600	4
HELPFULNESS	115251	5000	25000	2
RCT20K	180040	30212	30135	5

Table 2: Specification of task datasets. C refers to the number of classes. CHEMPROT (Kringelum et al., 2016) and RCT20K (Dernoncourt and Lee, 2017) are from biomedical domain. HYPERPARTISAN (Kiesel et al., 2019) and AGNEWS (Zhang et al., 2015) are from news domain. HELPFULNESS (McAuley et al., 2015) and IMDB (Maas et al., 2011) are from reviews domain. ACL-ARC (Jurgens et al., 2018) and SCIERC (Luan et al., 2018) are from CS domain. SST (Socher et al., 2013) is a general domain sentiment analysis task.

iments. It has 12 layers, 768 hidden dimensions per token, 12 attention heads and a total of 110 million parameters. It is pre-trained with a sum of two objectives. First is the masked language model objective where model learns to predict masked tokens. Second is the next sentence prediction objective where sentence learns to predict if sentence B follows sentence A or not. We use learning rate of 0.0001, batch size 256 and warm-up ratio 0.01. For finetuning, we pass the final layer [CLS] token embedding through a task specific feed-forward layer for prediction. We use learning rate 3e-5, batch size 8, warm-up ratio 0.1 and finetune the network for five epochs. In all the experiments, we start with a BERT pre-trained for one million steps and continue pre-training for additional $50,000$ steps in case of discrete λ, and $100,000$ steps in case of continuous λ. Also, as mentioned in Section 3.2, we filter out segments instead of sentences and save them. We set the segment length to be 15 sentences and filter out 20% of the data. Pseudo-code of the end-to-end algorithm can be found in Appendix B.

4.1 Baseline Methods

For each baseline data selection method, we start with a BERT pre-trained on general domain corpus for one million steps as in case of TADPOLE. Then, we continue pre-training the baseline method for the same number of steps as in case of our method. In case of baseline methods which filter general domain corpus, we filter the same fraction of text as in case of our method. Due to space constraints, we discuss some of the technical details of Baseline methods in Appendix **??**.

General: *Continued pre-training on general domain corpus.*

Random: *Continued pre-training on random subset of general domain corpus.*

Task (Gururangan et al., 2020): *Continued pre-training on task data.* Since task data is small, we can not pre-train on the task data for as many steps as in other cases. Instead we do 100 epochs, save the model after every epoch and pick the best one.

LM (Moore and Lewis, 2010): *Continued pre-training on text filtered via language models trained on task data.* We train two language models, one on the task data and another on a subset of general domain corpus (same size as the task data). We select sentences with lowest scores given by the function $f(s) = H_I(s) - H_O(s)$, where $H_I(s)$ and $H_O(s)$ are the cross-entropy between the n-gram distribution and the language model distribution.

Distance (Wang et al., 2017a): *Continued pre-training on data filtered via Euclidean distance scoring function.* For each sentence f, we consider BERT embedding v_f and compute vector centers $C_{F_{\text{in}}}$ and $C_{F_{\text{out}}}$ of the task data F_{in} and a random subset of general domain corpus F_{out}; $C_{F_{\text{in}}} = \frac{\sum_{f \in F_{\text{in}}} v_f}{|F_{\text{in}}|}, C_{F_{\text{out}}} = \frac{\sum_{f \in F_{\text{out}}} v_f}{|F_{\text{out}}|}$. We score a sentence f as per the scoring function:$\delta_f = d(v_f, C_{F_{\text{in}}}) - d(v_f, C_{F_{\text{out}}})$.

Data-Aug (Xie et al., 2019): *Data augmentation via back translation and tfidf based word replacement.* In back translation, for each sentnence f, we tranlate it to french and then back into english. In tfidf based word replacement, we replace uniformative words with other uninformative words. Label of the new training example is same as the label of original example. Due to space constraints, we report the average scores of the two data augmentation strategies. Note that data augmentation only applies to task data used while finetuning and does not involve additional pre-training.

4.2 Results

Table 3 shows the effectiveness of TADPOLE, automatically adapting pre-training to the task domain. Top half contains the result for four small datasets and bottom half contains the results for five large datasets. Since focus of the paper is on small datasets, we take subsamples of large datasets of size 500, 1000, 2000, 5000, 20000. For tasks with less than $5k$ samples, continuing pre-training on the unfiltered corpus (General or Random subset) yields an average gain less than 0.11%. Adapting pre-training by training on the task data only yields

Task	Base	General	Random	Task	LM	Distance	Data-Aug	TADPOLE	T+ Data-Aug
Small datasets: # training examples < 5K									
HPRPARTISAN	$70.57_{3.04}$	$70.97_{2.03}$	$71.04_{2.32}$	$70.88_{2.63}$	$71.47_{2.56}$	$72.16_{2.14}$	$71.18_{2.86}$	$73.58_{2.39}$	$\mathbf{73.87}_{2.86}$
ACL-ARC	$72.31_{4.7}$	$72.38_{3.93}$	$72.42_{3.71}$	$72.46_{3.48}$	$72.40_{1.85}$	$72.47_{2.64}$	$72.76_{3.03}$	$72.81_{3.83}$	$\mathbf{73.31}_{2.77}$
SCIERC	$82.84_{1.39}$	$82.85_{1.38}$	$82.81_{1.13}$	$83.18_{1.09}$	$82.99_{2.75}$	$83.40_{2.17}$	$83.27_{1.12}$	$\mathbf{85.85}_{0.95}$	$85.76_{0.95}$
CHEMPROT	$81.62_{0.74}$	$81.59_{0.67}$	$81.62_{0.71}$	$81.63_{0.82}$	$81.83_{0.74}$	$81.64_{0.76}$	$82.08_{0.88}$	$82.41_{0.62}$	$\mathbf{82.81}_{0.68}$
Average Gain	-	$0.11_{0.05}$	$0.15_{0.05}$	$0.20_{0.04}$	$0.34_{0.08}$	$0.34_{0.06}$	$0.48_{0.02}$	$1.82_{0.3}$	$\mathbf{2.1}_{0.26}$

Average gain over five tasks: SST, RCT20K, HELPFULNESS, IMDB, AGNews with subset of training samples

#Training samples	General	Random	Task	LM	Distance	Data-Aug	TADPOLE	T + Data-Aug
500	$0.2_{0.11}$	$0.21_{0.03}$	$0.18_{0.05}$	$0.22_{0.13}$	$0.4_{0.02}$	$0.21_{0.09}$	$\mathbf{2.16}_{0.06}$	$2.01_{0.13}$
1000	$0.04_{0.15}$	$-0.22_{0.07}$	$0.13_{0.05}$	$-0.24_{0.07}$	$0.18_{0.03}$	$0.24_{0.06}$	$2.01_{0.07}$	$\mathbf{2.04}_{0.04}$
2000	$0.14_{0.09}$	$0.16_{0.02}$	$0.23_{0.03}$	$0.18_{0.04}$	$0.2_{0.01}$	$0.1_{0.05}$	$1.48_{0.06}$	$\mathbf{1.52}_{0.03}$
5000	$0.09_{0.06}$	$0.14_{0.02}$	$0.11_{0.01}$	$0.0_{0.01}$	$0.08_{0.01}$	$0.12_{0.06}$	$1.0_{0.02}$	$\mathbf{1.31}_{0.05}$
20000	$0.08_{0.05}$	$0.07_{0.03}$	$0.0_{0.01}$	$-0.16_{0.03}$	$0.0_{0.01}$	$0.05_{0.05}$	$0.74_{0.09}$	$\mathbf{0.8}_{0.04}$

Table 3: Performance of TADPOLE, six baseline methods and TADPOLE combined with Data Augmentation. At top, we list the performance of each task whereas at bottom we list the average performance gain over five tasks such that for each task, we subsample a fixed number of training samples. Base corresponds to the pre-trained model on general domain corpus with no further pre-training. Baseline methods are mentioned in previous subsection. TADPOLE corresponds to our method with discrete relevance weights. T+Data-Aug corresponds to our method combined with data augmentation during finetuning. Keeping in line with the previous works, we use the following metrics: accuracy for SST, micro f1 score for CHEMPROT and RCT20K, macro f1 score for ACL-ARC, SCIERC, HELPFULNESS, HPRPARTISAN, IMDB, and AGNEWS. Each model is finetuned eight times with different seeds and the mean value is reported. Subscript correspond to the standard deviation in the finetuned model performance. Average gain corresponds to the average improvement over Base for each of the baseline methods and TADPOLE. Subscript in Average Gain corresponds to the standard deviation in the estimate of the average gain.

an average gain of 0.20%. Applying popular data selection methods known for domain adaptation including Language Model based relevance score or Distance based relevance score yields a maximum gain of 0.34%. TADPOLE beats all these methods and achieve an average gain of 1.82%. Data augmentation on task data while finetuning (no additional pre-training) achieves an average gain 0.48%. Combining Data-Augmentation with TADPOLE yields the maximum average gain showing that TADPOLE and Data Augmentation are complimentary methods and TADPOLE is superior to Data Augmentation on it's own. For tasks with more than $5k$ samples, we achieve an average gain of 0.36%. Detailed results can be found in Appendix G. To further test the efficacy of TADPOLE with small number of training samples, we randomly select a subset of training samples from the large task datasets and treat it as a new task. We observe that the gap between performance gain from TADPOLE and baseline methods increase when the number of training samples are low.

Models pre-trained on each of the four domain specific corpus can achieve a higher gain (3.37%) over the base model. However, unlike these models, our method has the advantage that it does not require access to any large domain specific corpus. Instead we only need a small task datasetalready available for finetuning. So, it is applicable to any new task from any new domain. We observe that Performance boost is higher if the corresponding boost via additional pre-training on large domain specific corpus is higher. Results for this comparison can be found in Appendix F. In Table 3, results are presented for the discrete relevant weight case as they are better when the number of steps available to continue pre-training are small. Results for continuous weights case can be found in Appendix E. Results are not very sensitive to the fraction of data filtered as can be seen in Figure 6 in Appendix H.

5 Conclusion

Domain shift in finetuning from Pre-training can significantly impact the performance of deep learning models. We address this issue in the most reasonable setting when we only have access to the labeled task data for finetuning. We adapt data selection methods from Domain Adaptation to adapt pre-training for the downstream task. The existing methods either require sufficiently large task data, or are based on adhoc techniques that do not generalize well across tasks. Our major contribution is providing a new data selection technique that performs well even with very little task data, and generalizes well across tasks.

References

Steven Abney. 2007. *Semisupervised learning for computational linguistics*. CRC Press.

Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, et al. 2018. Construction of the literature graph in semantic scholar. *arXiv preprint arXiv:1805.02262*.

Dogu Araci. 2019. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*.

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*.

Eyal Ben-David, Carmel Rabinovitz, and Roi Reichart. 2020. Perl: Pivot-based domain adaptation for pre-trained deep contextualized embedding models. *Transactions of the Association for Computational Linguistics*, 8:504–521.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. 2000. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104.

Raghavendra Chalapathy and Sanjay Chawla. 2019. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407*.

Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58.

Xia Cui and Danushka Bollegala. 2019. Self-adaptation for unsupervised domain adaptation. *Proceedings-Natural Language Processing in a Deep Learning World*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Franck Dernoncourt and Ji Young Lee. 2017. Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts. *arXiv preprint arXiv:1710.06071*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kevin Duh, Graham Neubig, Katsuhito Sudoh, and Hajime Tsukada. 2013. Adaptation data selection using neural language models: Experiments in machine translation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 678–683.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030.

Xiaoyi Gu, Leman Akoglu, and Alessandro Rinaldo. 2019. Statistical analysis of nearest neighbor methods for anomaly detection. In *Advances in Neural Information Processing Systems*, pages 10923–10933.

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. *arXiv preprint arXiv:2004.10964*.

Fouzi Harrou, Farid Kadri, Sondes Chaabane, Christian Tahon, and Ying Sun. 2015. Improved principal component analysis for anomaly detection: Application to an emergency department. *Computers & Industrial Engineering*, 88:63–77.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

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

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.

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.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. *arXiv preprint arXiv:1808.09602*.

Andrew 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.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 43–52.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Tri-Dzung Nguyen and Roy E Welsch. 2010. Outlier detection and robust covariance estimation using mathematical programming. *Advances in data analysis and classification*, 4(4):301–334.

Caleb C Noble and Diane J Cook. 2003. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *Proceedings of the 19th international conference on World wide web*, pages 751–760.

Nanyun Peng and Mark Dredze. 2016. Multi-task domain adaptation for sequence tagging. *arXiv preprint arXiv:1608.02689*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Jason Phang, Phu Mon Htut, Yada Pruksachatkun, Haokun Liu, Clara Vania, Katharina Kann, Iacer Calixto, and Samuel R Bowman. 2020. English intermediate-task training improves zero-shot cross-lingual transfer too. *arXiv preprint arXiv:2005.13013*.

Barbara Plank and Gertjan Van Noord. 2011. Effective measures of domain similarity for parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1566–1576.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Robert Remus. 2012. Domain adaptation using domain similarity-and domain complexity-based instance selection for cross-domain sentiment analysis. In *2012 IEEE 12th international conference on data mining workshops*, pages 717–723. IEEE.

Sebastian Ruder and Barbara Plank. 2017. Learning to select data for transfer learning with bayesian optimization. *arXiv preprint arXiv:1707.05246*.

Bernhard Schölkopf, Robert C Williamson, Alex J Smola, John Shawe-Taylor, and John C Platt. 2000. Support vector method for novelty detection. In *Advances in neural information processing systems*, pages 582–588.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Vincent Van Asch and Walter Daelemans. 2010. Using domain similarity for performance estimation. In *Proceedings of the 2010 Workshop on Domain Adaptation for Natural Language Processing*, pages 31–36.

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. *arXiv preprint arXiv:1708.00712*.

Rui Wang, Andrew Finch, Masao Utiyama, and Eiichiro Sumita. 2017a. Sentence embedding for neural machine translation domain adaptation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 560–566.

Rui Wang, Masao Utiyama, Lemao Liu, Kehai Chen, and Eiichiro Sumita. 2017b. Instance weighting for neural machine translation domain adaptation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1488.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, pages 9054–9065.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

Appendix

A Related Work

Domain Adaptation: A typical set up for Domain Adaptation involves access to labeled data in source domain, very limited or no labeled data in the target domain and unlabeled data in both source and target domains. This is somewhat different than the setup for our paper where we have access to labeled data with no additional unlabeled data in the task domain and our objective is optimize performance for the same domain. Nevertheless, several techniques of Domain Adaptation have similarities or core components useful for our setup. There are two sets of approaches addressing Domain Adaptation problem: model-centric and data-centric. Model-centric approaches redesign parts of the model: the feature space, the loss function or regularization and the structure of the model (Blitzer et al., 2006; Pan et al., 2010; Ganin et al., 2016). A recent such approach, appropriate for our setting is called Pivot-based Domain Adaptation; it has recently been applied to Task Adaptive Pre-training when there is additional unlabeled task data available (Ben-David et al., 2020). In a nutshell, the idea is to distinguish between pivot and non-pivot features, where pivot features behave similarly in both domains. Then, by converting the non-pivot to pivot features, one can make use of a model trained on the source data. This approach does not work well when the target data is small since the mapping of non-pivot to pivot features cannot be trained with a limited size dataset. Since our technique is data-centric and applies to the regime of a small target corpus, we do not further analyze this or any other model-centric approach.

Data-centric approaches for domain adaptation include pseudo-labeling, using auxiliary tasks and data selection. Pseudo-labeling apply a trained classifier to predict labels on unlabeled instances which are then treated as 'pseudo' gold labels for further training (Abney, 2007; Cui and Bollegala, 2019). Auxiliary-task domain adaptation use labeled data from auxiliary tasks via multi-task learning (Peng and Dredze, 2016) or intermediate-task transfer (Phang et al., 2018, 2020). The methods most relevant to us are those of data selection and are discussed above in detail.

B Datasets in accuracy estimation of Anomaly score based data filtration

CS task data: To train anomaly score discriminator for CS data, we use the tasks data from ACL-ARC and SCIERC. Details of these datasets are mentioned in Section 4.
CS and Bio Abstracts: Semantic Scholar corpus (Ammar et al., 2018) contains datasets from a variety of domain. We filter out text based on the domain field and only keep the abstracts from CS and bio domain.
News: We use REALNEWS (Zellers et al., 2019) corpus containing news articles from 500 news domains indexed by Google News. It is obtained by scraping dumps from Common Crawl.
Finance: We use the TRC2-financial dataset. This a subset of Reuters TRC24 corpus containing news articles published between 2008 and 2010. It can be obtained by applying here: https://trec.nist.gov/data/reuters/reuters.html

C Isolation Forest (Liu et al., 2008)

. Isolation Forest is an unsupervised decision tree ensemble method that identifies anomalies by isolating outliers of the data. It isolates anomalies in data points instead of profiling the normal points. Algorithm works by recursively partitioning the data using a random split between the minimum and maximum value of a random feature. It works due to the observation that outliers are less frequent than the normal points and lie further away from normal points in the feature space. Thus, in a random partitioning, anomalous points would require fewer splits on features resulting in shorter paths and distinguishing from the rest of the points. Anomaly score of a point x is defined as $s(x, n) = 2^{-\frac{E(h(x))}{c(n)}}$, where $E[h(x)]$ is the expected path length of x in various decision trees, $c(n) = 2H(n-1) - 2(n-1)/n$ is the average path length of unsuccessful search in a Binary Tree and $H(n-1)$ is the $n-1$-th harmonic number and n is the number of external nodes.

D Pseudo Code

Algorithm 1 shows the pseudo code for the case of continuous relevance weights. Discrete relevance weight setting is same as $C \to \infty$. As discussed in 3.2, in case of discrete relevance weights, we filter out segments containing several consec-

Algorithm 1 Task Adaptive Pre-training

Input: Pre-trained model B, Pre-training instances $x_1, \ldots, x_N$, task data $\mathcal{T}$, (C, α), #steps

Stage 1: Instance weight computation

Let the sentences of the task be $s_1, \ldots, s_t$ with sentence embeddings $P = \{\text{Embed}(s_1), \ldots, \text{Embed}(s_t)\}$.

Let a random subset of pre-training instances (sentences of these instances) be $s'_1, \ldots, s'_{t/10}$ with BERT based sentence embeddings $N = \{\text{Embed}(s'_1), \ldots, \text{Embed}(s'_{t/10})\}$

Train an anomaly detection object, IF = IsolationForest$(P \cup N)$

For $i \in [N]$, let $S(x_i) = \text{IF.score}(\text{Embed}(x_i))$

Let $\mu = \frac{1}{N}\sum_{i=1}^{N} S(x_i)$ and $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(S(x_i) - \mu)^2}$.

For every $i \in [N]$, $\bar{S}(x_i) = \frac{S(x_i)-\mu}{\sigma}$.

For every $i \in [N]$, $\lambda(x_i) = \frac{1}{1+e^{-C(\alpha - \bar{S}(x_i))}}$

Stage 2: Adaptation of pre-training to target domain

Continue training language model B for #steps on instances $x_1, \ldots, x_N$ with instance weights $\lambda(x_1), \ldots, \lambda(x_N)$.

Finetune resulting model on the labeled task data $\mathcal{T}$

utive sentences. We experimented with several options for the segment length and found the stable segment length to be 15 sentences. Here, a sentence is a consecutive piece of text such that when applied through the BERT tokenizer, it results in 256 sentences.

E Continuous relevance weights

We see in Table 4 that a model additionally pre-trained for 50,000 with discrete λ values consistently over performs the continuous case even when we train with continuous relevance weights for far higher number of steps. This is because of the fact that many of those steps yield virtually no training at all. For instance, suppose the relevance weights are uniformly distributed between 0 and 1; $[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]$. Then, in discrete case we pick the top two sentences and thus two steps are sufficient to train on these most relevant sentences (assume batch size is 1). However, in continuous case, we need to train the model for ten steps to train on these top two relevant sentences. Thus, we need many more steps to achieve and beat the performance achieved in the Discrete case. An open question is to combine the two settings so as to benefit from the generality of Continuous case and efficiency of the discrete case.

F Performance boost with Domain-specific Corpus vs TADPOLE

We compare the performance boost we achieved due to TADPOLE with the performance boost we achieve if we have access to large pre-training corpus. In Table 5, we list the gain in performance in both cases over eight tasks from four domains. We see that the performance boost is higher with TADPOLE if the corresponding boost is higher with domain specific corpus. Thus if there is a large domain shift between the general domain corpus and the task data, as can be measured by the performance boost via large pre-training corpus, then TADPOLE is able to achieve large performance boost via Task Adaptation. Scale of numbers in the two columns are not directly comparable due to the following two reasons. First is that additional pre-training done is Gururangan et al. (2020) is for almost as many steps as the number of steps required to pre-train a network from scratch. However, in our case additional pre-training is done for only 5% of the number of steps required to pre-train a network from scratch. Second reason is that the model used in (Gururangan et al., 2020) is different, RoBERTa. Also, the general domain corpus is different and thus the domain shift is not exactly the same as in our case. The point however remains the same, which is that as the target domain is further away from the pre-training corpus, the benefits of TADPOLE increase.

G Performance gain for large datasets

Table 6 shows the results for datasets with more than $5k$ training samples.

H Different data fraction

Figure 6 shows that results shown in Table 3 are not very sensitive to the fraction of pre-training data filtered. We can chose anywhere between 2-20% of the data.

Task	Base	Discrete	Continuous-1	Continuous-3
CHEMPROT	$81.62_{0.74}$	$\mathbf{82.41}_{0.62}$	$81.74_{0.81}$	$81.64_{0.83}$
RCT20K	$87.52_{0.16}$	$\mathbf{87.82}_{0.13}$	$87.49_{0.28}$	$87.56_{0.22}$
HPRPARTISAN	$70.57_{3.04}$	$\mathbf{73.58}_{2.39}$	$70.94_{1.98}$	$71.29_{2.95}$
AGNEWS	$93.99_{0.13}$	$\mathbf{94.03}_{0.16}$	$94.01_{0.14}$	$94.01_{0.15}$
HELPFULNESS	$69.30_{0.60}$	$\mathbf{69.70}_{0.92}$	$69.35_{0.5}$	$69.37_{0.44}$
IMDB	$88.65_{0.24}$	$\mathbf{89.29}_{0.22}$	$88.63_{0.51}$	$88.71_{0.46}$
ACL-ARC	$72.31_{4.7}$	$\mathbf{72.81}_{3.83}$	$72.26_{2.33}$	$72.36_{2.12}$
SCIERC	$82.84_{1.39}$	$\mathbf{85.85}_{0.95}$	$83.14_{1.96}$	$83.13_{2.65}$
SST	$92.02_{0.29}$	$\mathbf{92.42}_{0.32}$	$92.11_{0.32}$	$92.13_{0.37}$

Table 4: Comparison of discrete vs continuous relevance weight setting. Base corresponds to the pre-trained model on general domain corpus with no further pre-training. Discrete refers to TADPOLE with discrete relevance weights/filtered out text and pre-trained additionally for 50000 steps. Continuous-x refers to TADPOLE with continuous relevance weights and pre-trained additionally for $x * 100,000$ more steps. Metrics used for different tasks: accuracy for SST, micro f1 score for CHEMPROT and RCT20K, macro f1 score for ACL-ARC, SCIERC, HELPFULNESS, HPRPARTISAN, IMDB, and AGNEWS. Each model is finetuned eight times with different seeds and the mean value is reported. Subscript correspond to the standard deviation in the finetuned model performance.

Task	TADPOLE	Domain Corpus
CHEMPROT	0.79	2.3
RCT20K	0.3	0.4
HYPERPARTISAN	3.01	1.6
AGNEWS	0.04	0.0
HELPFULNESS	0.4	1.4
IMDB	0.64	5.4
ACL-ARC	0.5	3.5
SCIERC	3.01	12.4

Table 5: Performance boost via TADPOLE vs pre-training on domain specific corpus. TADPOLE corresponds to our method with discrete relevance weights/filtered out text and pre-trained additionally for 50000 steps. Domain Corpus refers to the model trained in (Gururangan et al., 2020) over the domain same as the downstream task. Metrics used for different tasks: accuracy for SST, micro f1 score for CHEMPROT and RCT20K, macro f1 score for ACL-ARC, SCIERC, HELPFULNESS, HPRPARTISAN, IMDB, and AGNEWS. Each model is finetuned eight times with different seeds. We report the difference in the mean value of performance between the model with additional pre-training and base model with no additional pre-training.

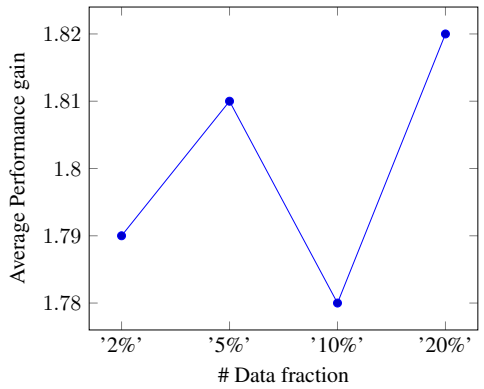

Figure 6: Performance gain as fraction of data filtered for tasks with less than $5k$ training samples.

Task	Base	General	Random	Task	LM	Distance	Data-Aug	TADPOLE	T+ Data-Aug
Large datasets: # training examples $\geq$ 5K									
IMDB	$88.65_{0.24}$	$88.53_{0.27}$	$88.63_{0.26}$	$88.77_{0.39}$	$88.67_{0.44}$	$88.69_{0.47}$	$88.75_{0.52}$	$\mathbf{89.29}_{0.22}$	$88.91_{0.17}$
SST	$92.02_{0.29}$	$92.21_{0.31}$	$92.14_{0.24}$	$92.21_{0.24}$	$92.25_{0.4}$	$92.15_{0.35}$	$91.98_{0.12}$	$92.42_{0.32}$	$\mathbf{92.67}_{0.55}$
AGNEWS	$93.99_{0.13}$	$94.06_{0.19}$	$94.09_{0.11}$	$94.04_{0.08}$	$94.03_{0.13}$	$94.04_{0.11}$	$93.97_{0.28}$	$94.03_{0.16}$	$\mathbf{94.45}_{0.3}$
HELPFULNESS	$69.30_{0.60}$	$69.39_{0.78}$	$69.34_{0.58}$	$69.41_{0.50}$	$69.58_{0.59}$	$69.42_{0.69}$	$69.3_{0.56}$	$\mathbf{69.70}_{0.92}$	$69.33_{0.64}$
RCT20K	$87.52_{0.16}$	$87.57_{0.16}$	$87.54_{0.17}$	$87.60_{0.18}$	$\mathbf{87.85}_{0.23}$	$87.62_{0.24}$	$87.57_{0.19}$	$87.82_{0.13}$	$87.83_{0.37}$
Average Gain	-	$0.02_{0.02}$	$0.04_{0.01}$	$0.11_{0.01}$	$0.18_{0.03}$	$0.09_{0.01}$	$0.02_{0.01}$	$\mathbf{0.36}_{0.04}$	$0.34_{0.04}$

Table 6: Performance of TADPOLE and five Baseline methods. Base corresponds to the pre-trained model on general domain corpus with no further pre-training. Baseline methods are mentioned in previous subsection. TADPOLE corresponds to our method with discrete relevance weights. T+Data-Aug corresponds to our method combined with data augmentation during finetuning. Keeping in line with the previous works, we use the following metrics: accuracy for SST, micro f1 score for CHEMPROT and RCT20K, macro f1 score for ACL-ARC, SCIERC, HELPFULNESS, HPRPARTISAN, IMDB, and AGNEWS. Each model is finetuned eight times with different seeds and the mean value is reported. Subscript correspond to the standard deviation in the finetuned model performance. Average gain corresponds to the average improvement over Base for each of the baseline methods and TADPOLE. Subscript in Average Gain corresponds to the standard deviation in the estimate of the average gain.