International Journal of Advanced Medical Sciences and Technology (IJAMST)
ISSN: 2582-7596 (Online), Volume-4 Issue-6, October 2024

ULMFiT: Universal Language Model Fine-Tuning for Text Classification

Shenson Joseph, Herat Joshi

Abstract: ***While inductive transfer learning has revolutionized computer vision, current approaches to natural language processing still need training from the ground up and task-specific adjustments. As a powerful transfer learning approach applicable to any NLP activity, we provide Universal Language Model Fine-tuning (ULMFiT) and outline essential strategies for language model fine-tuning. With an error reduction of 18–24% on most datasets, our technique considerably surpasses the state-of-the-art on six text categorization tasks. Additionally, it achieves the same level of performance as training on 100 times more data with only 100 annotated examples. We have made our pretrained models and code publicly available.***

Keywords: ULMFiT, Learning, Code, Language, NLP, Techniques, Strategies

I. INTRODUCTION

The ULMFiT method, which is like fine-tuning ImageNet models, makes it possible to do robust inductive transfer learning for any natural language processing (NLP) task. It also fixes the issues with fine-tuning-based inductive transfer, like the need for large datasets and the time it takes to converge. The same 3-layer LSTM architecture with the same hyperparameters and no changes other than tweaking the dropout hyperparameters did better than highly designed models and transfer learning techniques on six well-researched text classification tasks [1].

While inductive transfer learning has revolutionized computer vision, current methods in natural language processing still need tailoring training to individual tasks [2,3]. As a powerful transfer learning approach applicable to any NLP work, we provide Universal Language Model Fine-tuning (ULMFiT) and outline essential strategies for language model fine-tuning. An LM is a probability distribution over word sequences. Text categorization, summarization, text creation, and other natural language processing tasks employ this language model as a foundational model, similar to transfer learning [4,5]. Computer vision (CV) has been greatly influenced by inductive transfer learning [6].

Manuscript received on 02 October 2024 | Revised Manuscript received on 11 October 2024 | Manuscript Accepted on 15 October 2024 | Manuscript published on 30 October 2024.
*****Correspondence Author(s)
Herat Joshi*, Department of Analytics & Decision Support, Great River Health Systems, Burlington, (Iowa), United States of America (USA). Email ID: heratjoshi@gmail.com, ORCID ID: 0009-0009-4199-544X
Shenson Joseph, Department of Computer Engineering, University of North Dakota, Houston, (Texas), United States of America (USA). Email ID: shenson.joseph@gmail.com, ORCID ID: 0009-0001-5191-5556

Object identification, classification, and segmentation models used in applied CV are usually fine-tuned versions of models that were pre-trained on datasets like ImageNet and MS-COCO. Text classification, a subset of Natural Language Processing (NLP) activities, encompasses commercial document classification, including for legal discovery, emergency response, spam, fraud, and bot detection, among other real-world applications [7,8].

Deep learning models have surpassed human performance on several natural language processing (NLP) tasks; nevertheless, they must be trained from the beginning, which requires massive datasets and a convergence time of days [9,10]. A basic transfer strategy that just addresses a model's first layer, fine-tuning pre-trained word embeddings (inductive transfer), has had a substantial effect in practice and is employed by most state-of-the-art models [11,12]. Context mapping becomes much simpler using LMs. Like an uncle to an aunt or a king to a queen, a man-to-woman map is a formality. The joint probability distribution among words is the result of these mappings. A statistical likelihood matrix (LM) is one term for this [13]. Statistical language modeling, often known as language modeling or simply LM, is the process of creating probabilistic models that can, given a sequence of words, predict the word that will come after it [14].

II. REVIEW OF LITERATURE

S. Min (2021) [15] The weights of several tasks are shared in multi-task designs. So, the know-how for various jobs is pooled. Hyper columns, a new approach that use multi-level embeddings, go beyond transfer pretrained word embeddings. Another way to transfer learning in natural language processing is called fine-tuning. This technique involves training a model for one task and then adjusting it to perform better on the target job. As shown, the fine-tuning method in natural language processing achieves respectable outcomes when it comes to the transfer of knowledge across comparable tasks.

Deng, (2022) [16] Because it allows to train models in a completely different issue with less data, transfer learning is essential for deep learning applications. The field of computer vision makes extensive use of transfer learning. For the most part, the model will learn the picture characteristics from a generic job, such as object recognition, using the ImageNet dataset, and then apply them to other domains. Some of these tasks include refining a pre-trained model's final layer or layers and then applying that knowledge in new settings.

The research was conducted to examine how the complexity of datasets affects transfer learning using CNN.

III. OBJECTIVES

1. To introduce ULMFiT as a general transfer learning method for NLP into areas where there is no fine-tuning or specific fine-tuning is difficult.
2. To explain the techniques that are the most important for the final tuning of a language model and which are the basic skills for using ULMFiT for the ready-made solution to a variety of NLP tasks.

IV. PROBLEM STATEMENT

The study focuses on the challenges inherent in the application of transfer learning for text classification in the field of NLP. The study seeks to address the challenge in using existing transfer learning approaches such as supervised pretraining tasks to develop better ways for generalizing in scenarios with limited training data or resources. Specifically, the authors identify three settings where language model fine-tuning could be particularly useful: 1) Developing NLP systems for non-English languages with limited training data 2) New NLP tasks with no or little establishment of the state-of-the-art arch I 3) Some labeled data available but limited data available.

V. SIGNIFICANCE OF THE STUDY

The importance of the study on ULMFiT (Universal Language Model Fine-tuning for Text Classification) is attributed to the fact that it can help to further develop NLP and applying it to various disciplines. The results obtained by the study show that state-of-the-art performance on widely used text classification tasks can be reached by applying language model fine-tuning as a transfer learning method. Among other things the study suggests that ULMFiT may be especially valuable in the following cases: For languages other than English, as well as for new NLP tasks with yet unseen architectures, or tasks in which labeled data is scarce. This makes transfer learning feasible in more real-world scenarios and facilitates positive contributions to under-resourced domains. Finally, the study presents future research opportunities in further optimizing language model pretraining and fine-tuning; leveraging the approach for new downstream tasks; and understanding the knowledge learned by pretrained models. The following possibilities for future research may be used to further improve and apply for the performance and efficiency of language model fine-tuning methods. In summary, the study's importance is in the achievement of state-of-the-art text classification with the application of transferable data-scarce scenario approaches in NLPL.

VI. RESEARCH METHODOLOGY

Our focus is on the broadest possible context for inductive transfer learning in natural language processing: We want to improve performance on every target task fT given static source task fS and any static task fS that is equal to or greater than fT. As an alternative to ImageNet in natural language processing, language modeling might be considered an excellent source task: It catches emotion, hierarchical linkages, and long-term relationships, all of which are important for downstream activities. "Unlike MT and entailment, it yields almost infinite data for the majority of domains and languages. Furthermore, we demonstrate that a pretrained LM may be readily adjusted to the peculiarities of a target job, leading to a significant improvement in performance. Language modeling is also an integral part of current activities like MT and conversation engineering. The hypothesis space H, which is inferred from language modeling, could be helpful for a wide variety of natural language processing problems. We present ULMF Tuning, an approach that builds a language model (LM) using a huge general-domain corpus for pretraining and then uses innovative techniques to fine-tune it for the target job. Because it satisfies these realistic requirements, the approach is applicable everywhere: 1) It is applicable to tasks with different document sizes, numbers, and label types; 2) It needs no new feature engineering or preprocessing; 3) It does not require additional in-domain documents or labels; and 4) It employs a single architecture and training method.

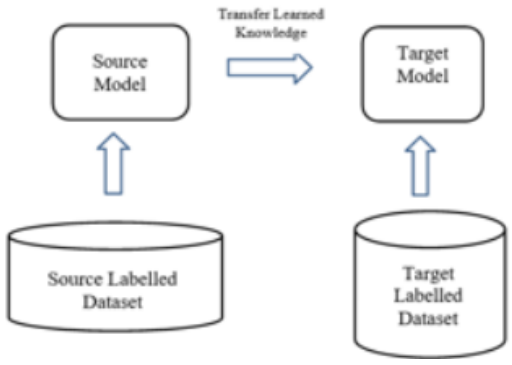

Fig. 1: Architecture for UMLFiT

For our tests, we use the cutting-edge AWD-LSTM language model, which is essentially a standard LSTM with a number of tunable dropouts hyperparameters but without attention, short-cut connections, or any other advanced features. We anticipate that, similar to CV, further use of higher-performance language models will lead to greater downstream performance. As shown in Figure 1, ULMFiT is comprised of the following steps: 3.1) Pretraining for general-domain LM; 3.2) Fine-tuning for target task LM; and 3.3) Fine-tuning for target task classifier. In what follows, we'll go over these points.

A. LM Pretraining for General Domains

A big, general-property-capturing corpus of language, similar to ImageNet, would be ideal. We use Wikitext-103, a dataset of 28,595 preprocessed Wikipedia articles and 103 million words, to pretrain the language model. Tasks with small datasets benefit the most from pretraining, which allows generalization even with 100 labeled samples. We anticipate that more varied pretraining corpora might improve performance, but we will leave that investigation for future study.

The costliest part of the process, this step enhances downstream model performance and convergence and only has to be done once.

B. Fine-Tuning of the Target Task LM

While pretraining with various general-domain data is important, the target task data will most likely have a distinct distribution. So, we train the LM on task-specific data to make it more accurate. With a pretrained general-domain LM as a starting point, this step can train a robust LM with less data, and it converges quicker since it just has to adjust to the target data's peculiarities. In order to fine-tune the LM, we provide the following: discriminative fine-tuning and slanted triangular learning rates.

Customized discrimination Because many layers gather different kinds of data, they should all be fine-tuned to varying degrees. This is why we provide discriminative fine-tuning3, a new approach to fine-tuning.

Using discriminative fine-tuning, we may adjust the learning rates of the model's layers independently, rather than applying a uniform rate to all of them. Just to give you an idea, this is how the parameters θ of a model are updated regularly using stochastic gradient descent (SGD) at time step t:

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla_\theta J(\theta)$$

In this context, η represents the learning rate and ∇θJ(θ) signifies the gradient in relation to the objective function of the model. The parameters θ are divided into {θ1,.., ŘL} for discriminative fine-tuning, with θl being the model parameters at the l-th layer and L being the number of layers in the model. Similarly, for each l-th layer, we get {η1,..., ŷL}, where ŷl represents the learning rate. Consequently, here is the SGD update that includes discriminative fine-tuning:

$$\theta_t^l = \theta_{t-1}^l - \eta^l \cdot \nabla_{\theta^l} J(\theta)$$

Based on our empirical findings, we recommend fine-tuning the final layer to determine its learning rate ŷL, and then utilizing ŷl−1 = ŷl/2.6 for the lower levels. Rates of skewed triangular learning At the outset of training, we aim for the model to converge to an appropriate area of the parameter space so that it may be fine-tuned to task-specific characteristics thereafter. To do this, it is not recommended to use an annealed learning rate or a constant learning rate (LR) throughout training. Slanted triangular learning rates (STLR) are an alternative that we suggest. They follow the following schedule of updates: initially, they linearly raise the learning rate; subsequently, they linearly decay it.

$$cut = \lfloor T \cdot cut\_frac \rfloor$$

$$p = \begin{cases} t/cut, & \text{if } t < cut \\ 1 - \frac{t-cut}{cut \cdot (ratio-1)}, & \text{otherwise} \end{cases}$$

$$\eta_t = \eta_{max} \cdot \frac{1 + p \cdot (ratio - 1)}{ratio}$$

where T is the total number of training iterations, cut frac is the fraction of iterations where the LR is increased, cut is the iteration when the LR is decreased, and p is the proportion of iterations where the LR is increased or decreased, respectively. ηt represents the learning rate at iteration t, and the ratio describes the distance from the greatest LR ηmax to the lowest LR. In most cases, we use cut frac = 0.1, ratio = 32, and ηmax = 0.01.

STLR significantly improves performance by adjusting triangle learning rates with a brief rise and a lengthy decay period. We compare aggressive co-sine annealing, a comparable schedule that has recently been used to attain state-of-the-art performance in CV.

C. Optimizing the Target Task Classifier

As a last step in classifier tuning, we add two more linear blocks to the pretrained language model. A softmax activation that produces a probability distribution across target classes at the final layer and ReLU activations for the intermediate layer follow conventional practice for CV classifiers. Batch normalization and dropout are also used in each block. It should be noted that the only parameters that are learnt from scratch are those in these task-specific classifier layers. The states of the pooled final hidden layer are sent into the first linear layer.

D. Sharing Resources

When it comes to text categorization jobs, the signal is usually only a few of words scattered across the document. Considering only the most recent hidden state of the model might lead to information loss, especially when input documents can include hundreds of words. We combine the hidden state at the latest time step hT of the document with the max-pooled and mean-pooled representations of the hidden states across as many time steps as can be stored in GPU memory, H = {h1,..., hT }, since this is why.

$$\mathbf{h}_c = [\mathbf{h}_T, \mathtt{maxpool}(\mathbf{H}), \mathtt{meanpool}(\mathbf{H})]$$

The most important aspect of the transfer learning approach is fine-tuning the target classifier. If you tighten the screws too much, you'll end up with catastrophic forgetting and all the useful data from language modeling will be lost. On the other hand, if you tighten the screws too little, you'll end up with sluggish convergence and overfitting. We suggest progressive unfreezing as an additional method for classifier fine-tuning alongside discriminative fine-tuning and triangular learning rates. Slow defrosting We recommend unfreezing the model in stages, beginning with the last layer—which contains the least general knowledge—rather than fine-tuning all layers simultaneously, as this could lead to catastrophic forgetting. After unfreezing the last layer, we fine-tune all unfrozen layers for one epoch. We continue this process of fine-tuning each layer until convergence is achieved at the final iteration, after which we unfreeze the next lower frozen layer. Just as in "chain-thaw", we train a single layer at a time, but in this case we add more layers to the collection of "thawed" layers and keep going.

In Section 5, we demonstrate how the three components—discriminative fine-tuning, slanted triangular learning rates, and progressive unfreezing—work together to improve our method's performance on various datasets.

To facilitate gradient propagation for long input sequences, BPTT for Text Classification (BPT3C) language models are trained using BPTT. We suggest BPTT for Text Classification (BPT3C) to make classifier fine-tuning for big texts possible: We split the document into batches of size b, which are fixed in length. We remember the hidden states for max-pooling and mean-pooling, and we start each batch by loading the model with the previous batch's final state. variations in gradient.

Table 1: Text Classification Datasets or Tasks with Number of Classes and Training Instances

Dataset	Type	# Classes	# Examples
TREC-6	Question	6	5.5k
IMDb	Sentiment	2	25k
Yelp-bi	Sentiment	2	560k
Yelp-full	Sentiment	5	650k
AG	Topic	4	120k
DBpedia	Topic	14	560k

A two-way language model Just with previous research, we aren't confined to honing a language model that just works in one way. We pretrain a forward and a backward LM for each experiment. Before averaging the classifier predictions, we use BPT3C to separately fine-tune a classifier for each LM.

E. Experiments

Although our method may be used for sequence labeling as well, text classification challenges are the main focus of our study because of the practical importance of these tasks.

F. Experimental Environment

Tasks and datasets We test our approach on six popular datasets that have been used by top-tier text classification and transfer learning methods for three common text classification tasks: sentiment analysis, question classification, and topic classification. The datasets range in document length and number of documents. In Table 1, we provide the statistics for every dataset and job.

Evaluating Public Opinion, the binary IMDb movie review dataset and the binary and five-class versions of the Yelp review dataset are used to assess our method for sentiment analysis.

Classification of Questions We make use of the six-class variant of the tiny TREC dataset, which consists of fact-based queries that are open-domain and organized into broad semantic categories.

Table 2: Test Error Rates (%) on Two Text Classification Datasets

Model (IMDb)	Test	Model (TREC—6)	Test
CoVe [17]	8.2	CoVe [17]	4.2
oh-LSTM [22]	5.9	TBCNN [19]	4.0
Virtual l [18]	5.9	LSTM-CNN [20]	3.9
ULMFiT (ours)	4.6	ULMFiT (ours)	3.6

Subject categorization [21] generated the large-scale AG news and DBpedia ontology datasets, which we use for topic categorization.

Initial preparation [18] and [17] both used the identical pre-processing that we do here. Furthermore, we provide unique tokens for capitalization, elongation, and repetition to enable the language model to grasp features that might be pertinent for categorization. Critical parameters We are looking for a model that can reliably do a variety of jobs. We adjust the same set of hyperparameters on the IMDb validation set across jobs, unless otherwise stated. The AWD-LSTM language model that we use has a BPTT batch size of 70, three layers, 1,150 hidden activations per layer, and an embedding size of 400. Layers are subjected to a 0.4 dropout, RNN layers to a 0.3 dropout, input embedding layers to a 0.4 dropout, embedding layers to a 0.05 dropout, and the RNN hidden-to-hidden matrix to a 0.5 weight dropout. There is a 50-size hidden layer in the classifier. In place of the default values of 0.9 and 0.99, we use Adam with $\beta 1 = 0.7$, which is comparable to. When fine-tuning the LM and classifier, we utilize 64-bit batches, 0.004 as the base learning rate and 0.01 as the classifiers. We also adjust the number of epochs on the validation set for each job. Aside from that, we follow the same procedures as in. Starting points and models for comparison We check our results against the state-of-the-art for every job. We evaluate our results using CoVe, a cutting-edge NLP transfer learning algorithm, using the IMDb and TREC-6 datasets. We test our results against the most advanced text classification algorithm on the AG, Yelp, and DBpedia datasets.

VII. RESULTS AND DISCUSSION

We report each result as an error rate, preferring lower numbers to ensure uniformity. The test error rates used by [17] on the IMDb and TREC-6 datasets are provided in Table. On both datasets, our strategy outperforms the state-of-the-art and CoVe, a hypercolumn-based state-of-the-art transfer learning method. We achieve a 43.9% reduction in inaccuracy on IMDb when compared to CoVe and a 22% reduction when compared to the state of the-art. Our technique utilizes a conventional LSTM with dropout, offering a potential improvement over the current state-of-the-art, which involves complicated architectures, numerous types of attention, and advanced embedding strategies [17,22].

We see that our method obtains an error of 4.6 on IMDb, whereas the language model fine-tuning strategy only manages an error of 7.64. This highlights the advantage of using our fine-tuning techniques to transfer information from a big ImageNet-like corpus. Among these datasets, IMDb serves as a representative example: Many business applications, including product response monitoring and support email routing, rely on sentiment analysis. Sentiment analysis typically produces documents that are only a few lines long, such as emails used in legal discovery or online comments used in community management.

The modest size of the 500-example test set means that our improvement, like those of state-of-the-art techniques, is not statistically significant on TREC-6. Although our model may handle cases ranging from a single line in TREC-6 to many paragraphs in IMDb, the competitive performance on TREC-6 shows that it works well across dataset sizes.

Despite our pretraining data being approximately two orders of magnitude less than their 7 million phrase pairings, we consistently outperformed [17] on both datasets.

In the table, we present the test error rates for the three larger datasets: AG, DBpedia, and Yelp-full. Once again, our approach significantly outperforms the state-of-the-art. We find a similarly striking 23.7% decrease in errors on AG when compared to the state-of-the-art. Our mistake reduction rates on DBpedia, Yelp-bi, and Yelp-full are 4.8%, 18.2%, and 2.0%, respectively. We conduct several studies and ablations to determine the relative importance of each contributor. We conduct our tests on three diverse corpora, namely IMDb, TREC-6, and AG,

Table 3: Rates of Test Errors on Datasets for Text Categorization

	AG	DBpedia	Yelp-bi	Yelp-full
Char-level CNN [21]	9.51	1.55	4.88	37.95
CNN [22]	6.57	0.84	2.9	32.39
DPCNN [23]	6.87	0.88	2.64	30.58
ULMFiT (ours)	5.01	0.8	2.16	29.98

which include a range of activities, genres, and sizes. We use 10% of the training set for each experiment's validation set and present the error rates on this set using unidirectional LMs. Training all algorithms with the exception of ULMFiT with early halting, we fine-tune the classifier for 50 epochs.

A. Minimalist Instruction

The ability to train a model with a limited number of labels is a major advantage of transfer learning. We test ULMFiT on two scenarios: one where all task data is accessible and may be used for LM fine-tuning ('semi-supervised'), and another where only labeled samples are used ('supervised'). We evaluate ULM-FiT in comparison to hyper column-based methods that need starting from zero during training. Using the same hyperparameters and a balanced data split for training, we hold the validation set constant.

Supervised ULMFiT on IMDb has the same performance as training from start with 10 times more data, whereas on AG it equals the performance with 20 times more data, proving that general-domain LM pretraining is beneficial. At 100 labeled examples, we get the same performance as training from scratch with 50 times more data on AG and 100 times more data on IMDb, respectively, if we let ULMFiT to additionally use unlabeled examples (50k for IMDb and 100k for AG). With fewer and shorter instances, both supervised and semi-supervised ULMFiT perform similarly on TREC-6, where ULMFiT shows a considerable improvement after initial training.

B. Results From Pretraining

With respect to WikiText-103, we contrast the use of pretraining and no pretraining in Fig. Small and medium-sized datasets, the most typical in commercial applications, are ideal for pretraining. The performance benefits of pretraining are evident even for huge datasets.

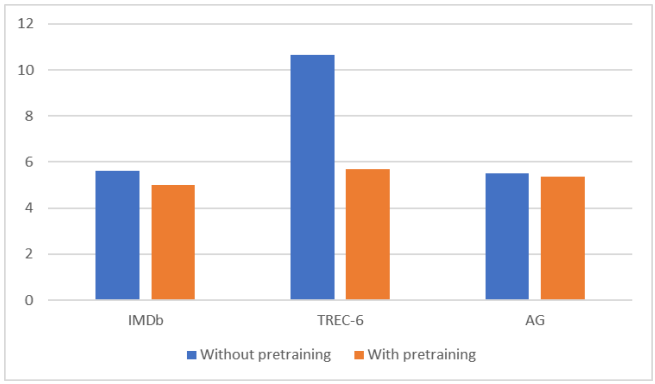

Fig. 2: Pretraining and Non-Pretraining ULMFiT Validation Error Rates

C. The Effect of LM Caliber

Performance on the bigger datasets using our fine-tuning approaches. A vanilla LM without dropout on the smaller TREC-6 could be overfit, leading to worse performance.

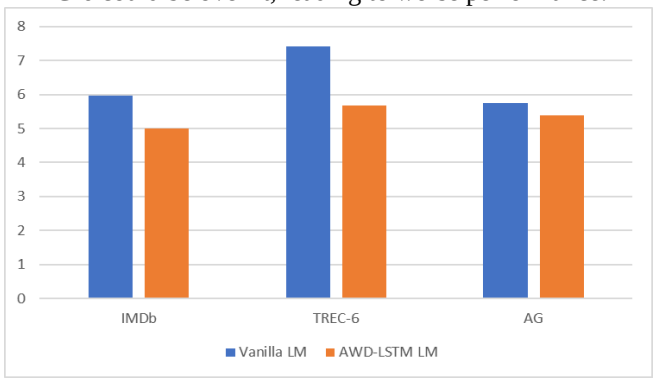

Fig. 3: The AWD-LSTM LM and a Vanilla LM Validate ULMFiT with Different Error Rates

D. Consequences of Optimizing LM

The most popular fine-tuning strategy, fine-tuning the complete model ('complete'), is compared in Fig, along with two other methods: slanted triangular learning rates ('Stlr') and discrimini-native fine-tuning ('Discr'). Large datasets are ideal for LM fine-tuning. On the smaller TREC-6, where frequent fine-tuning is not advantageous, 'Discr' and 'Stlr' are essential for improving performance across all three datasets.

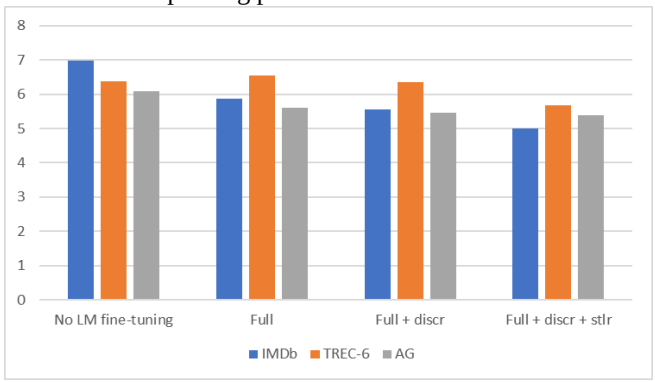

Fig. 4: Error Rates in ULMFiT Validation Using Various LM Fine-Tuning Methods

E. The Effect of Fine-Tuning the Classifier

From the ground up, we examine four different approaches: "Full" for fine-tuning the whole model, "Last" for fine-tuning only the final layer, "Chain-thaw," and "Freez" for progressive unfreezing. We go on to evaluate the significance of slanted triangle learning rates ('Stlr') and discriminative fine-tuning ('Discr'). We evaluate the second one in relation to a different, more intensive cosine annealing schedule ('Cos'). For the 'Discr' layer, we use a learning rate of ηL = 0.01. For the 'Chain-thaw' layer, we use a learning rate of 0.001 for all layers except the final one, and a learning rate of 0.001 for the last layer. The outcomes are shown in Fig

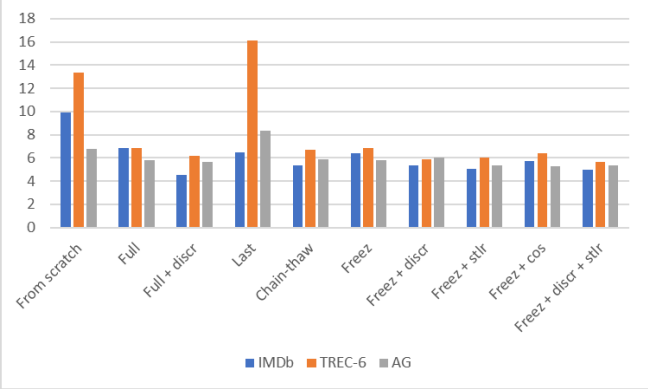

Fig. 5: Ratios of Validation Errors for ULMFiT Using Various Techniques to Refine the Classifier

On the little TREC-6 in particular, fine-tuning the classifier yields far better results than starting from scratch during training. As a result of its significant underfitting, the conventional fine-tuning procedure in CV, "Last," is unable to achieve a training error of zero. On smaller datasets, "chain-thaw" performs competitively, but it is vastly outdone on the huge AG. As with "Full," "Freez" delivers comparable performance. With the exception of the big AG, 'Discr' always improves 'Full' and 'Freez' performance. Even with slanted triangular learning rates, cosine annealing competes well on big datasets but fails miserably on smaller ones. Finally, the best performance on IMDB and TREC-6, as well as competitive performance on AG, are achieved with complete ULMFiT classifier fine-tuning (bottom row). Most importantly, ULMFiT is the only global approach as it consistently produces outstanding results.

F. Classifier Behavior While Fine-Tuning

We found that classifier fine-tuning makes a big effect, yet in natural language processing, fine-tuning for inductive transfer is still mainly ignored since it's believed to be useless. Figure shows a comparison of the classifier's validation error during training with that of the ULMFiT and 'Full' fine-tuned versions to help us understand our model's fine-tuning behavior.

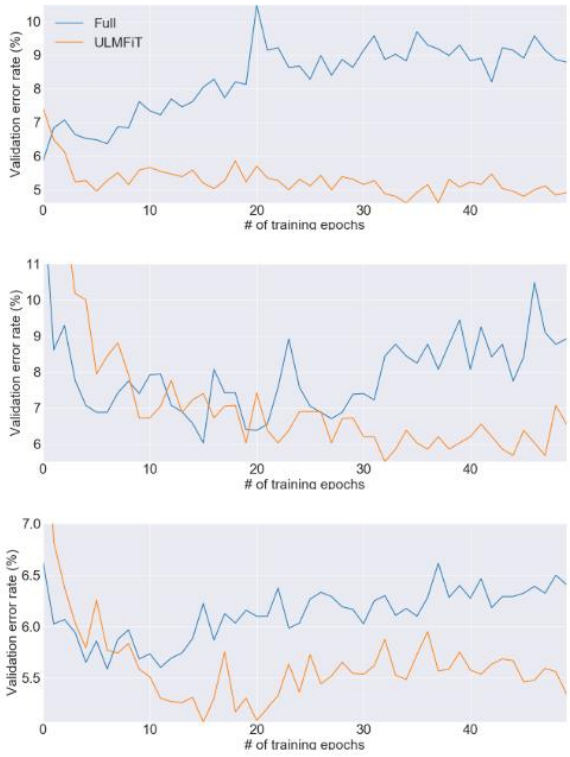

Fig. 6: The Classifier's Validation Error Rate curves while using ULMFiT and 'Full' on AG, TREC-6, and IMDb

The comparably lowest error occurs early in training while fine-tuning the whole model, regardless of the dataset (for example, after the first epoch on IMDb).

As the model begins to overfit and loses the information gained from pretraining, the inaccuracy grows. The learning rate schedule has a beneficial impact on ULMFiT, which is more stable and does not experience catastrophic forgetting; performance stays the same or even improves until late epochs.

Effects of directionality transfer Combining the predictions of a forward and backward LM-classifier yields a performance improvement of around 0.5-0.7, but it comes at the expense of training a second model. By using the bidirectional model on IMDb, we were able to reduce the test error from 5.30 for the single model to 4.58.

VIII. DISCUSSION

We have shown that ULMFiT can outperform state-of-the-art methods on popular text classification problems. However, we think that fine-tuning language models will be more effective than current transfer learning strategies in the following scenarios. a) Natural language processing (NLP) for languages other than English, where there is a dearth of training data for supported pretraining tasks; b) novel NLP tasks in the absence of a state-of-the-art architecture; and c) tasks involving small quantities of labeled and unlabeled data [24].

Given the lack of research into transfer learning and, in particular, NLP fine-tuning, several potential future approaches exist [25,26]. The prediction of a subset of words, such as the most common ones, might retain most of the performance while speeding up training, according to recent work that shows that an alignment be-tween source and target task label sets is important. This could be one way to improve language model pretraining and fine-tuning, making them more scalable [27,28]. For example, ImageNet only experiences a small performance drop when predicting fewer classes. In a multi-task learning approach, more jobs may be added to language models, or they can be enhanced with extra supervision, for example syntax-sensitive dependencies." Develop a weakly-supervised model that maintains its universal qualities; this model should be either more generic or more suited to specific downstream tasks [29].

Exploring new tasks and models using the approach is another potential avenue to pursue. Although extending sequence labeling is a piece of cake, tasks like entailment or question answering, which involve more complicated interactions, may call for creative pretraining and fine-tuning strategies. Lastly, while we have shown a number of analyses and ablations, more research is needed to fully understand the information that a pretrained language model takes in, how this changes when it is fine-tuned, and what data is needed for certain jobs [30][31][32][33].

IX. CONCLUSION

Any natural language processing (NLP) job may benefit from our proposed ULMFiT, a transfer learning strategy that is both effective and extremely sample-efficient. We have also suggested a number of new methods for fine-tuning that, when used together, may avoid catastrophic forgetting and allow for strong learning on a variety of tasks. On six benchmark text categorization problems, our approach blew away competing transfer learning methods and the current state of the art. We anticipate that our findings will spur more advancements in natural language processing transfer learning.

A. Findings of the Study

The author has demonstrated that ULMFiT is capable of performing state-of-the-art text classification for common tasks. The researcher believes that language model fine-tuning will be particularly useful in the following settings compared to existing transfer learning approaches: b. Natural Language Processing (NLP) for languages where supervised pretraining tasks have limited data. b. New NLP tasks in which no state-of-the-art architecture is available. c. Low-resource machine-learning tasks with small amounts of labeled data and large amounts of unlabeled data.

The researcher suggests that future directions could include a further scaling up of the language model pretraining and fine-tuning, e.g., predicting only partial words (e.g., most frequent words) or employing multi-task learning or other supervision. b. The method could be applied to new tasks or novel models such as sequence labeling, entailment, or question answering, which may require innovative approaches to pretrain and fine-tune. c. Perform more experiments to investigate how much knowledge a pre-trained model contains, how it evolves during various fine-tuning processes, and what knowledge is needed for different tasks. Therefore, the researcher acknowledges that despite conducting analyses and adjustments for the pretrained language model and fine-tuning, additional studies are still required.

The study shows the efficiency of ULMFiT for text classification tasks and provides a new perspective on the use of the language model fine-tuning for a variety of NLP use cases with small data or resources. The authors also suggest several further research areas to advance the method and to elucidate its theoretical foundation.

B. Scope for Further Research

Future work can focus on investigating language model pretraining and fine-tuning strategies in the process of improving their scalability. This can include predicting a part of the most frequent words, employing multi-task learning, or adding more supervision. Another potential direction to explore is developing new pretraining and fine-tuning strategies to extend the application of ULMFiT to sequence labeling, entailment, and question answering. To gain a deeper understanding of the knowledge in pretrained language models, its changes during fine-tuning, and the specific types of information required for various tasks, it is crucial to conduct extensive analyses. It would be fascinating to consider the performance of ULMFiT with limited training data for non-English languages and further understand the ULMFiT potential for transfer learning across languages. Investigating ULMFiT's synergies and gains with other transfer methods is another option. Using ULMFiT to test how well transfer learning works in low-resource settings with little labeled data and finding out if the amount of unlabeled data affects its performance are also interesting directions to pursue further.

DECLARATION STATEMENT

After aggregating input from all authors, I must verify the accuracy of the following information as the article's author.

- **Conflicts of Interest/ Competing Interests:** Based on my understanding, this article has no conflicts of interest.
- **Funding Support:** This article has not been sponsored or funded by any organization or agency. The independence of this research is a crucial factor in affirming its impartiality, as it has been conducted without any external sway.
- **Ethical Approval and Consent to Participate:** The data provided in this article is exempt from the requirement for ethical approval or participant consent.
- **Data Access Statement and Material Availability:** The adequate resources of this article are publicly accessible.
- **Authors Contributions:** The authorship of this article is contributed equally to all participating individuals.

REFERENCES

1. Adeborna, Esi and Siau, Keng, "AN APPROACH TO SENTIMENT ANALYSIS –THE CASE OF AIRLINE QUALITY RATING"(2014).

PACIS 2014 Proceedings. Paper 363. http://aisel.aisnet.org/pacis2014/363
2. Hung T. Vo, Hai C. Lam, Duc Dung Nguyen, Nguyen Huynh Tuong, "Topic classification and sentiment analysis for Vietnamese education survey system. (2016). In Asian Journal of Computer Science and Information Technology (Vol. 6, Issue 3). Innovative Journal. https://doi.org/10.15520/ajcsit.v6i3.44 .
3. Sarkar, S., Seal, T., & Bandyopadhyay, S. K. (2016). *Sentiment analysis - An objective view*. Journal for Research, 2(2), 26-29. https://www.researchgate.net/publication/328610677_Sentiment_Analysis-An_Objective_View
4. Joseph, Shenson and Joshi, Herat and Hassan, Md. Mehedi and Bairagi, Anupam Kumar, Advancing Quantum Machine Learning: From Theoretical Concepts to Experimental Implementations (July 01, 2024). Available at SSRN: https://ssrn.com/abstract=4946682 or http://dx.doi.org/10.2139/ssrn.4946682
5. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. J Mach Learn Res 15, 1929-1958. http://jmlr.org/papers/volume15/srivastava14a/srivastava14a.pdf
6. Joshi, H. (2022). Navigating the intersection of machine learning and healthcare: A review of current applications. International Journal of Advanced Research in Computer and Communication Engineering, 11(10). https://doi.org/10.17148/IJARCCE.2022.111016
7. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. https://www.deeplearningbook.org/
8. Zagoruyko, S., & Komodakis, N. (2016). Wide Residual Networks. Proceedings of the British Machine Vision Conference 2016. doi:10.5244/c.30.87 DOI: https://dx.doi.org/10.5244/C.30.87
9. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE. https://doi.org/10.1109/cvpr.2016.90
10. Howard, J., & Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics. https://doi.org/10.18653/v1/p18-1031
11. Joshi, H., Joseph, S., & Shukla, P. (2024). Unlocking Potential. In Advances in Medical Technologies and Clinical Practice (pp. 313–341). IGI Global. https://doi.org/10.4018/979-8-3693-5893-1.ch016
12. M. E. Peters, W. Ammar, C. Bhagavatula, and R. Power, "Semisupervised sequence tagging with bidirectional language models," arXiv preprint arXiv:1705.00108, 2017. https://doi.org/10.18653/v1/p17-1161
13. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., & Darrell, T. (2013). DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition (Version 1). arXiv. https://doi.org/10.48550/ARXIV.1310.1531
14. Long, J., Shelhamer, E., & Darrell, T. (2014). Fully Convolutional Networks for Semantic Segmentation (Version 2). arXiv. https://doi.org/10.48550/ARXIV.1411.4038
15. Sarhan, I., & Spruit, M. (2020). Can We Survive without Labelled Data in NLP? Transfer Learning for Open Information Extraction. In Applied Sciences (Vol. 10, Issue 17, p. 5758). MDPI AG. https://doi.org/10.3390/app10175758
16. Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, & Li Fei-Fei. (2009). ImageNet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition. 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPR Workshops). IEEE. https://doi.org/10.1109/cvpr.2009.5206848
17. Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in Translation: Contextualized Word Vectors. In Advances in Neural Information Processing Systems, 2017. https://doi.org/10.48550/arXiv.1708.00107
18. McCann, B., Bradbury, J., Xiong, C., & Socher, R. (2017). Learned in Translation: Contextualized Word Vectors (Version 2). arXiv. https://doi.org/10.48550/ARXIV.1708.00107
19. Mou, L., Peng, H., Li, G., Xu, Y., Zhang, L., & Jin, Z. (2015). Discriminative Neural Sentence Modeling by Tree-Based Convolution (Version 5). arXiv. https://doi.org/10.48550/ARXIV.1504.01106 .
20. Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., & Xu, B. (2016). Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling (Version 1). arXiv. https://doi.org/10.48550/ARXIV.1611.06639
21. Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level Convolutional Networks for Text Classification (Version 3). arXiv. https://doi.org/10.48550/ARXIV.1509.01626
22. Johnson, R., & Zhang, T. (2016). Supervised and Semi-Supervised Text Categorization using LSTM for Region Embeddings (Version 2). arXiv. https://doi.org/10.48550/ARXIV.1602.02373
23. Johnson, R., & Zhang, T. (2017). Deep Pyramid Convolutional Neural Networks for Text Categorization. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Association for Computational Linguistics. https://doi.org/10.18653/v1/p17-1052 .
24. Patel, V. A., & Joshi, M. V. (2018). Convolutional neural network with transfer learning for rice type classification. In J. Zhou, P. Radeva, D. Nikolaev, & A. Verikas (Eds.), Tenth International Conference on Machine Vision (ICMV 2017). Tenth International Conference on Machine Vision (ICMV 2017). SPIE. https://doi.org/10.1117/12.2309482 .
25. Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics. https://doi.org/10.3115/v1/d14-1162 .
26. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality (Version 1). arXiv. https://doi.org/10.48550/ARXIV.1310.4546 .
27. Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2016). Enriching Word Vectors with Subword Information (Version 2). arXiv. https://doi.org/10.48550/ARXIV.1607.04606 .
28. M. Rei, "Semi-supervised multitask learning for sequence labeling," arXiv preprint https://arxiv.org/pdf/1704.07156
29. Joshi, H. (2024). Artificial Intelligence in Project Management: A Study of The Role of Ai-Powered Chatbots in Project Stakeholder Engagement. In Indian Journal of Software Engineering and Project Management (Vol. 4, Issue 1, pp. 20–25). Lattice Science Publication (LSP). https://doi.org/10.54105/ijsepm.b9022.04010124
30. McCann, B., Bradbury, J., Xiong, C., & Socher, R. (2017). Learned in Translation: Contextualized Word Vectors (Version 2). arXiv. https://doi.org/10.48550/ARXIV.1708.00107
31. Kumbhakarna, V. M., Kulkarni, S. B., & Dhawale, A. D. (2020). NLP Algorithms Endowed f or Automatic Extraction of Information from Unstructured Free Text Reports of Radiology Monarchy. In International Journal of Innovative Technology and Exploring Engineering (Vol. 9, Issue 12, pp. 338–343). https://doi.org/10.35940/ijitee.l8009.1091220
32. Chellatamilan, T., Valarmathi, B., & Santhi, K. (2020). Research Trends on Deep Transformation Neural Models for Text Analysis in NLP Applications. In International Journal of Recent Technology and Engineering (IJRTE) (Vol. 9, Issue 2, pp. 750–758). https://doi.org/10.35940/ijrte.b3838.079220
33. Hudaa, S., Setiyadi, D. B. P., Lydia, E. L., Shankar, K., Nguyen, P. T., Hashim, W., & Maseleno, A. (2019). Natural Language Processing utilization in Healthcare. In International Journal of Engineering and Advanced Technology (Vol. 8, Issue 6s2, pp. 1117–1120). https://doi.org/10.35940/ijeat.f1305.0886s219

AUTHORS PROFILE

Herat Joshi is a distinguished Healthcare Technology and Informatics expert. He excels in driving technological advancements and innovation in Healthcare Information Management, Healthcare Informatics, and Medical/Bioinformatics. He has earned a PhD in Computer Science & Engineering with expertise in Data Science and AI, holds an MBA, and a bachelor's degree in computer engineering. He is associated with AMIA American Medical Informatics Association, AHIMA American Healthcare Information Management Association and IHA Iowa Health Association.

Shenson Joseph is a distinguished AI researcher and data science expert. With expertise in Data Science, Analytics, and Artificial Intelligence, he has authored 2 books and authored more than 6 research papers. Shenson has judged many national and international events and actively contributes to editorial boards and conferences. He has earned a master's degree in data science and a second master's degree in electrical & computer engineering. He is IEEE senior member and associated with ACM and AAAI (association for the advancement of artificial intelligence).