Universal Language Model Fine-tuning (ULMFiT) for Text Classification

ACL 2018 Paper

Yufeng Ma

Digital Library Research Laboratory
Computer Science
Virginia Tech

yufengma@vt.edu

September 20, 2018

Paper and Author Infos

Paper and Author Infos

arXiv:1801.06146v5 [cs.CL] 23 May 2018

Universal Language Model Fine-tuning for Text Classification

Jeremy Howard*
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder*
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io

Abstract

Inductive transfer learning has greatly impacted computer vision, but existing approaches in NLP still require task-specific modifications and training from scratch. We propose Universal Language Model Fine-tuning (ULMFiT), an effective transfer learning method that can be applied to any task in NLP, and introduce techniques that are key for fine-tuning a language model. Our method significantly outperforms the state-of-the-art on six text classification tasks, reducing the error by 18-24% on the majority of datasets. Furthermore, with only 100 labeled examples, it matches the performance of training from scratch on 100× more data. We open-source our pretrained models and code[1].

1 Introduction

Inductive transfer learning has had a large impact on computer vision (CV). Applied CV models (including object detection, classification, and segmentation) are rarely trained from scratch, but instead are fine-tuned from models that have been pretrained on ImageNet, MS-COCO, and other datasets (Sharif Razavian et al., 2014; Long et al., 2015a; He et al., 2016; Huang et al., 2017).

Text classification is a category of Natural Language Processing (NLP) tasks with real-world applications such as spam, fraud, and bot detection (Jindal and Liu, 2007; Ngai et al., 2011; Chu et al., 2012), emergency response (Caragea et al., 2011), and commercial document classification, such as for legal discovery (Roitblat et al., 2010).

While Deep Learning models have achieved state-of-the-art on many NLP tasks, these models are trained from scratch, requiring large datasets, and days to converge. Research in NLP focused mostly on *transductive* transfer (Blitzer et al., 2007). For *inductive* transfer, fine-tuning pretrained word embeddings (Mikolov et al., 2013), a simple transfer technique that only targets a model's first layer, has had a large impact in practice and is used in most state-of-the-art models. Recent approaches that concatenate embeddings derived from other tasks with the input at different layers (Peters et al., 2017; McCann et al., 2017; Peters et al., 2018) still train the main task model from scratch and treat pretrained embeddings as fixed parameters, limiting their usefulness.

In light of the benefits of pretraining (Erhan et al., 2010), we should be able to do better than *randomly initializing* the remaining parameters of our models. However, inductive transfer via fine-tuning has been unsuccessful for NLP (Mou et al., 2016). Dai and Le (2015) first proposed fine-tuning a language model (LM) but require millions of in-domain documents to achieve good performance, which severely limits its applicability.

We show that not the idea of LM fine-tuning but our lack of knowledge of how to train them effectively has been hindering wider adoption. LMs overfit to small datasets and suffered catastrophic forgetting when fine-tuned with a classifier. Compared to CV, NLP models are typically more shallow and thus require different fine-tuning methods.

We propose a new method, Universal Language Model Fine-tuning (ULMFiT) that addresses these issues and enables robust inductive transfer learning for any NLP task, akin to fine-tuning ImageNet models: The same 3-layer LSTM architecture—with the same hyperparameters and no additions other than tuned dropout hyperparameters—outperforms highly engineered models and trans-

[1] http://nlp.fast.ai/ulmfit.
*Equal contribution. Jeremy focused on the algorithm development and implementation, Sebastian focused on the experiments and writing.

Paper and Author Infos

Universal Language Model Fine-tuning for Text Classification

Jeremy Howard*
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder*
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io

Abstract

Inductive transfer learning has greatly impacted computer vision, but existing approaches in NLP still require task-specific modifications and training from scratch. We propose Universal Language Model Fine-tuning (ULMFiT), an effective transfer learning method that can be applied to any task in NLP, and introduce techniques that are key for fine-tuning a language model. Our method significantly outperforms the state-of-the-art on six text classification tasks, reducing the error by 18-24% on the majority of datasets. Furthermore, with only 100 labeled examples, it matches the performance of training from scratch on 100× more data. We open-source our pretrained models and code[1].

1 Introduction

Inductive transfer learning has had a large impact on computer vision (CV). Applied CV models (including object detection, classification, and segmentation) are rarely trained from scratch, but instead are fine-tuned from models that have been pretrained on ImageNet, MS-COCO, and other datasets (Sharif Razavian et al., 2014; Long et al., 2015a; He et al., 2016; Huang et al., 2017).

Text classification is a category of Natural Language Processing (NLP) tasks with real-world applications such as spam, fraud, and bot detection (Jindal and Liu, 2007; Ngai et al., 2011; Chu et al., 2012), emergency response (Caragea et al., 2011), and commercial document classification, such as for legal discovery (Roitblat et al., 2010).

[1] http://nlp.fast.ai/ulmfit.
*Equal contribution. Jeremy focused on the algorithm development and implementation, Sebastian focused on the experiments and writing.

While Deep Learning models have achieved state-of-the-art on many NLP tasks, these models are trained from scratch, requiring large datasets, and days to converge. Research in NLP focused mostly on *transductive* transfer (Blitzer et al., 2007). For *inductive* transfer, fine-tuning pretrained word embeddings (Mikolov et al., 2013), a simple transfer technique that only targets a model's first layer, has had a large impact in practice and is used in most state-of-the-art models. Recent approaches that concatenate embeddings derived from other tasks with the input at different layers (Peters et al., 2017; McCann et al., 2017; Peters et al., 2018) still train the main task model from scratch and treat pretrained embeddings as fixed parameters, limiting their usefulness.

In light of the benefits of pretraining (Erhan et al., 2010), we should be able to do better than *randomly initializing* the remaining parameters of our models. However, inductive transfer via fine-tuning has been unsuccessful for NLP (Mou et al., 2016). Dai and Le (2015) first proposed fine-tuning a language model (LM) but require millions of in-domain documents to achieve good performance, which severely limits its applicability.

We show that not the idea of LM fine-tuning but our lack of knowledge of how to train them effectively has been hindering wider adoption. LMs overfit to small datasets and suffered catastrophic forgetting when fine-tuned with a classifier. Compared to CV, NLP models are typically more shallow and thus require different fine-tuning methods.

We propose a new method, Universal Language Model Fine-tuning (ULMFiT) that addresses these issues and enables robust inductive transfer learning for any NLP task, akin to fine-tuning ImageNet models: The same 3-layer LSTM architecture—with the same hyperparameters and no additions other than tuned dropout hyperparameters—outperforms highly engineered models and trans-

SOTA on Text Classification

The 56th Annual Meeting of the Association for Computational Linguistics

- Jeremy Howard
- Sebastian Ruder

Paper and Author Infos

Universal Language Model Fine-tuning for Text Classification

Jeremy Howard*
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder*
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io

Abstract

Inductive transfer learning has greatly impacted computer vision, but existing approaches in NLP still require task-specific modifications and training from scratch. We propose Universal Language Model Fine-tuning (ULMFiT), an effective transfer learning method that can be applied to any task in NLP, and introduce techniques that are key for fine-tuning a language model. Our method significantly outperforms the state-of-the-art on six text classification tasks, reducing the error by 18-24% on the majority of datasets. Furthermore, with only 100 labeled examples, it matches the performance of training from scratch on 100× more data. We open-source our pretrained models and code[1].

1 Introduction

Inductive transfer learning has had a large impact on computer vision (CV). Applied CV models (including object detection, classification, and segmentation) are rarely trained from scratch, but instead are fine-tuned from models that have been pretrained on ImageNet, MS-COCO, and other datasets (Sharif Razavian et al., 2014; Long et al., 2015a; He et al., 2016; Huang et al., 2017).

Text classification is a category of Natural Language Processing (NLP) tasks with real-world applications such as spam, fraud, and bot detection (Jindal and Liu, 2007; Ngai et al., 2011; Chu et al., 2012), emergency response (Caragea et al., 2011), and commercial document classification, such as for legal discovery (Roitblat et al., 2010).

[1] http://nlp.fast.ai/ulmfit.
*Equal contribution. Jeremy focused on the algorithm development and implementation, Sebastian focused on the experiments and writing.

While Deep Learning models have achieved state-of-the-art on many NLP tasks, these models are trained from scratch, requiring large datasets, and days to converge. Research in NLP focused mostly on *transductive* transfer (Blitzer et al., 2007). For *inductive* transfer, fine-tuning pretrained word embeddings (Mikolov et al., 2013), a simple transfer technique that only targets a model's first layer, has had a large impact in practice and is used in most state-of-the-art models. Recent approaches that concatenate embeddings derived from other tasks with the input at different layers (Peters et al., 2017; McCann et al., 2017; Peters et al., 2018) still train the main task model from scratch and treat pretrained embeddings as fixed parameters, limiting their usefulness.

In light of the benefits of pretraining (Erhan et al., 2010), we should be able to do better than *randomly initializing* the remaining parameters of our models. However, inductive transfer via fine-tuning has been unsuccessful for NLP (Mou et al., 2016). Dai and Le (2015) first proposed fine-tuning a language model (LM) but require millions of in-domain documents to achieve good performance, which severely limits its applicability.

We show that not the idea of LM fine-tuning but our lack of knowledge of how to train them effectively has been hindering wider adoption. LMs overfit to small datasets and suffered catastrophic forgetting when fine-tuned with a classifier. Compared to CV, NLP models are typically more shallow and thus require different fine-tuning methods.

We propose a new method, Universal Language Model Fine-tuning (ULMFiT) that addresses these issues and enables robust inductive transfer learning for any NLP task, akin to fine-tuning ImageNet models: The same 3-layer LSTM architecture—with the same hyperparameters and no additions other than tuned dropout hyperparameters—outperforms highly engineered models and trans-

SOTA on Text Classification

The 56th Annual Meeting of the Association for Computational Linguistics

- Jeremy Howard
- Sebastian Ruder

Affiliations:

- fast.ai
- University of San Francisco

Paper and Author Infos

Universal Language Model Fine-tuning for Text Classification

Jeremy Howard[*]
fast.ai
University of San Francisco
j@fast.ai

Sebastian Ruder[*]
Insight Centre, NUI Galway
Aylien Ltd., Dublin
sebastian@ruder.io

Abstract

Inductive transfer learning has greatly impacted computer vision, but existing approaches in NLP still require task-specific modifications and training from scratch. We propose Universal Language Model Fine-tuning (ULMFiT), an effective transfer learning method that can be applied to any task in NLP, and introduce techniques that are key for fine-tuning a language model. Our method significantly outperforms the state-of-the-art on six text classification tasks, reducing the error by 18-24% on the majority of datasets. Furthermore, with only 100 labeled examples, it matches the performance of training from scratch on 100× more data. We open-source our pretrained models and code[1].

1 Introduction

Inductive transfer learning has had a large impact on computer vision (CV). Applied CV models (including object detection, classification, and segmentation) are rarely trained from scratch, but instead are fine-tuned from models that have been pretrained on ImageNet, MS-COCO, and other datasets (Sharif Razavian et al., 2014; Long et al., 2015a; He et al., 2016; Huang et al., 2017).

Text classification is a category of Natural Language Processing (NLP) tasks with real-world applications such as spam, fraud, and bot detection (Jindal and Liu, 2007; Ngai et al., 2011; Chu et al., 2012), emergency response (Caragea et al., 2011), and commercial document classification, such as for legal discovery (Roitblat et al., 2010).

[1]http://nlp.fast.ai/ulmfit.
*Equal contribution. Jeremy focused on the algorithm development and implementation, Sebastian focused on the experiments and writing.

While Deep Learning models have achieved state-of-the-art on many NLP tasks, these models are trained from scratch, requiring large datasets, and days to converge. Research in NLP focused mostly on *transductive* transfer (Blitzer et al., 2007). For *inductive* transfer, fine-tuning pretrained word embeddings (Mikolov et al., 2013), a simple transfer technique that only targets a model's first layer, has had a large impact in practice and is used in most state-of-the-art models. Recent approaches that concatenate embeddings derived from other tasks with the input at different layers (Peters et al., 2017; McCann et al., 2017; Peters et al., 2018) still train the main task model from scratch and treat pretrained embeddings as fixed parameters, limiting their usefulness.

In light of the benefits of pretraining (Erhan et al., 2010), we should be able to do better than *randomly initializing* the remaining parameters of our models. However, inductive transfer via fine-tuning has been unsuccessful for NLP (Mou et al., 2016). Dai and Le (2015) first proposed fine-tuning a language model (LM) but require millions of in-domain documents to achieve good performance, which severely limits its applicability.

We show that not the idea of LM fine-tuning but our lack of knowledge of how to train them effectively has been hindering wider adoption. LMs overfit to small datasets and suffered catastrophic forgetting when fine-tuned with a classifier. Compared to CV, NLP models are typically more shallow and thus require different fine-tuning methods.

We propose a new method, Universal Language Model Fine-tuning (ULMFiT) that addresses these issues and enables robust inductive transfer learning for any NLP task, akin to fine-tuning ImageNet models: The same 3-layer LSTM architecture— with the same hyperparameters and no additions other than tuned dropout hyperparameters— outperforms highly engineered models and trans-

SOTA on Text Classification

The 56th Annual Meeting of the Association for Computational Linguistics

- Jeremy Howard
- Sebastian Ruder

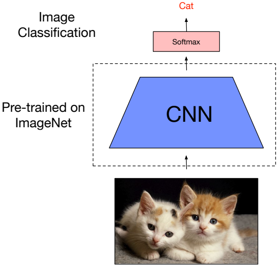
Affiliations:

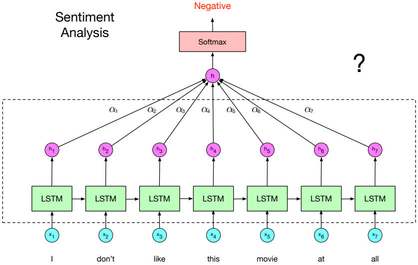
- fast.ai
- University of San Francisco
- Insight Centre, NUI Galway

Overview

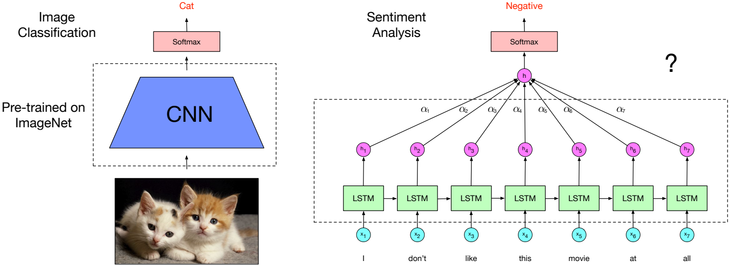
Introduction

Introduction



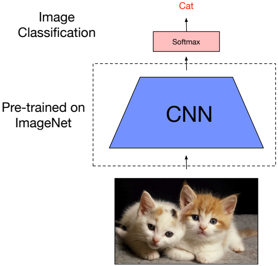


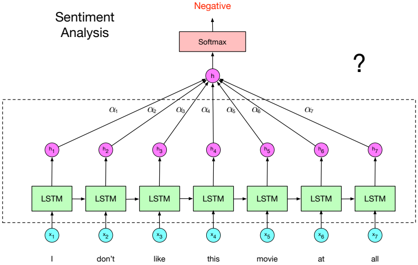
Introduction



- Computer Vision (CV) models rarely trained from scratch;

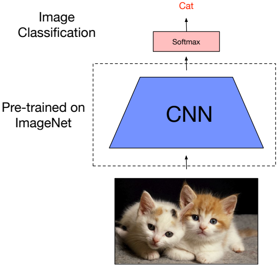
Introduction

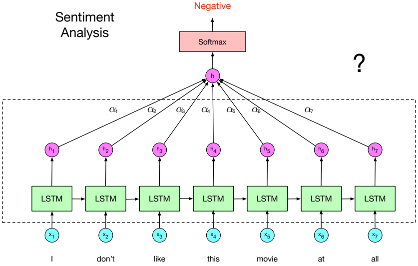




- Computer Vision (CV) models rarely trained from scratch;
- What can we fine-tune from for NLP tasks?

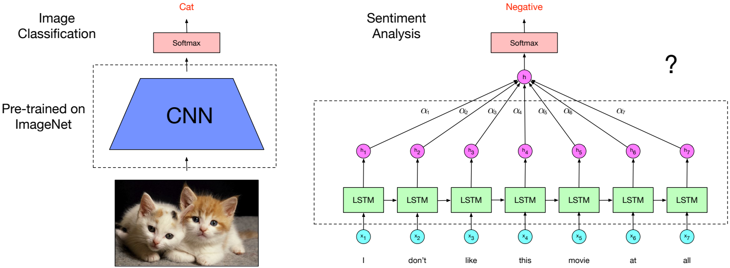
Introduction





- Computer Vision (CV) models rarely trained from scratch;
- What can we fine-tune from for NLP tasks?
- How to transfer learn from pretrained LM for NLP tasks?

Introduction



- Computer Vision (CV) models rarely trained from scratch;
- What can we fine-tune from for NLP tasks?
- How to transfer learn from pretrained LM for NLP tasks?

Motivation

How to design specific techniques to fine-tune LM effectively?

Related Work

Related Work

Transfer learning in CV

- General to task-specific [Yosinski et al. 2014]
- Transferring and fine-tune layers [Long et al. 2015, Sharif et al. 2014]

Related Work

Transfer learning in CV

- General to task-specific [Yosinski et al. 2014]
- Transferring and fine-tune layers [Long et al. 2015, Sharif et al. 2014]

Hypercolumns

- Hypercolumns [Hariharan et al. 2015]
- Go beyond word embeddings [Peters et al. 2017]

Related Work

Transfer learning in CV

- General to task-specific [Yosinski et al. 2014]
- Transferring and fine-tune layers [Long et al. 2015, Sharif et al. 2014]

Hypercolumns

- Hypercolumns [Hariharan et al. 2015]
- Go beyond word embeddings [Peters et al. 2017]

Multi-task learning

- Jointly train LM and main tasks [Rei et al. 2017, Liu et al. 2018]

Related Work

Transfer learning in CV

- General to task-specific [Yosinski et al. 2014]
- Transferring and fine-tune layers [Long et al. 2015, Sharif et al. 2014]

Hypercolumns

- Hypercolumns [Hariharan et al. 2015]
- Go beyond word embeddings [Peters et al. 2017]

Multi-task learning

- Jointly train LM and main tasks [Rei et al. 2017, Liu et al. 2018]

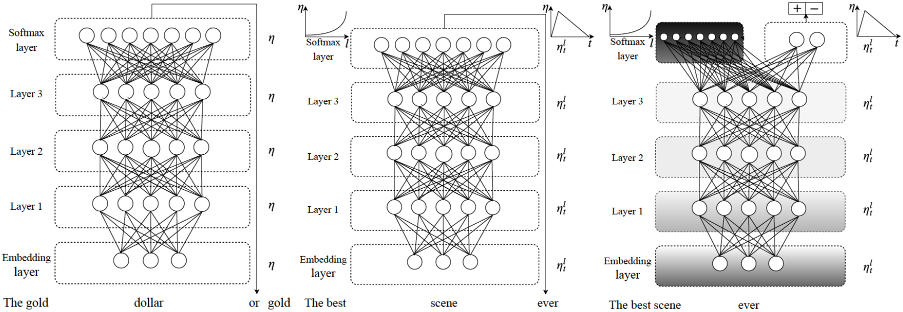
Fine-tuning

- Fine-tune LM, require lots of examples [Dai et al. 2015]

Overview

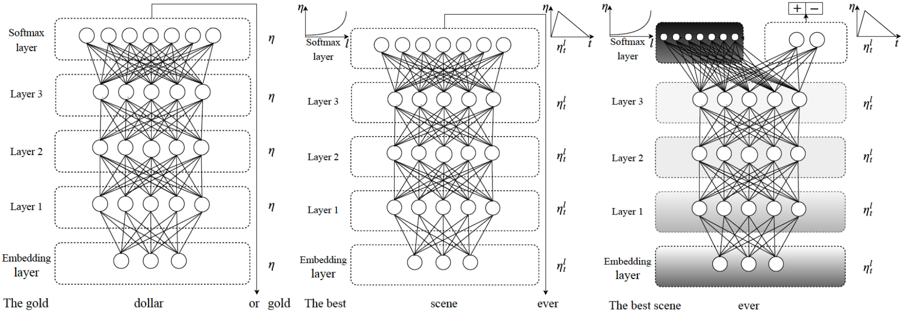
Universal Language Model Fine-tuning

Universal Language Model Fine-tuning



(a) LM pre-training (b) LM fine-tuning (c) Classifier fine-tuning

Universal Language Model Fine-tuning

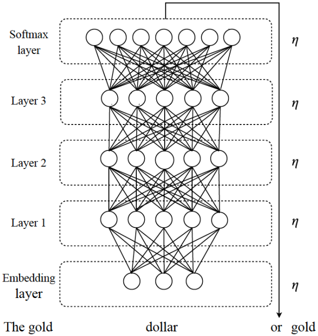


(a) LM pre-training (b) LM fine-tuning (c) Classifier fine-tuning

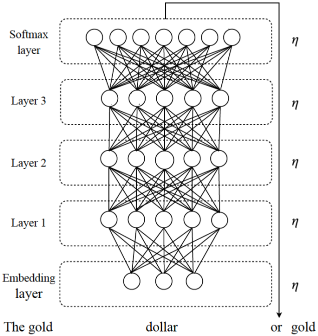
- LM ($\mathcal{T}_S$) $\rightarrow$ Text Classification ($\mathcal{T}_T$)
- Universal novel techniques for LM fine-tuning
- Based on AWD-LSTM [Merity et al. 2017]

General-domain LM pretraining

General-domain LM pretraining



General-domain LM pretraining

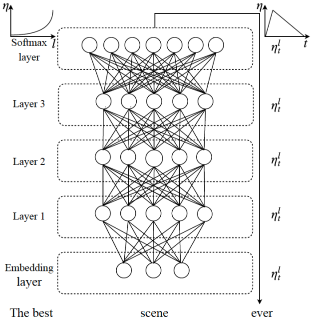


$\mathcal{T}_S$ LM pretraining

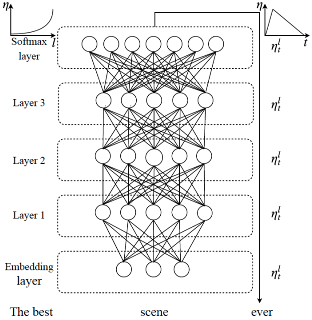
- $P(x_t|x_1, \ldots, x_{t-1})$
- Wikitext-103: $28,595$ articles and 103M words;

Target task LM fine-tuning

Target task LM fine-tuning



Target task LM fine-tuning



$\mathcal{T}_T$ LM pretraining

- Discriminative fine-tuning
- Slanted triangular learning rates

Target task LM fine-tuning

Target task LM fine-tuning

Discriminative fine-tuning

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

$$\eta^{l-1} = \eta^l / 2.6$$

Target task LM fine-tuning

Discriminative fine-tuning

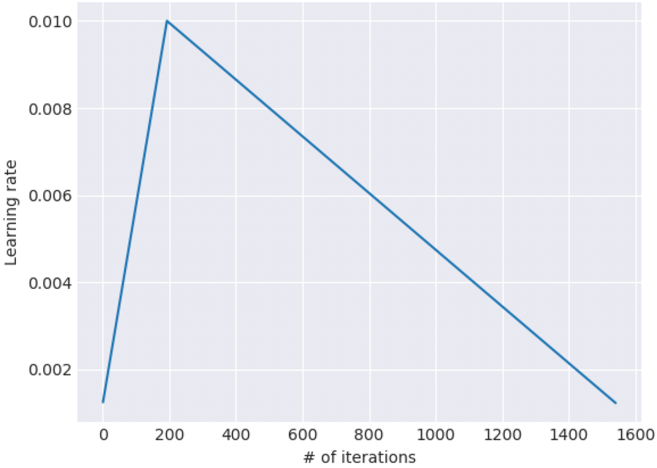
$$\theta_t^l = \theta_{t-1}^l - \eta^l \cdot \nabla_{\theta^l} J(\theta)$$
$$\eta^{l-1} = \eta^l / 2.6$$

Slanted triangular learning rates

$$cut = \lfloor T \cdot cut\_frac \rfloor$$
$$p = \begin{cases} t/cut, & \text{if } t < cut \\ 1 - \frac{t-cut}{cut \cdot (1/cut\_frac - 1)}, & \text{otherwise} \end{cases}$$
$$\eta_t = \eta_{max} \cdot \frac{1 + p \cdot (ratio - 1)}{ratio}$$

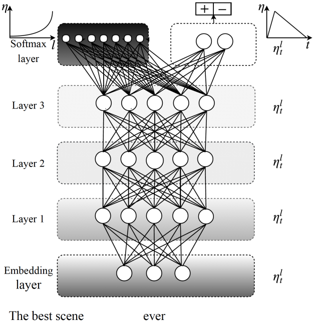
Slanted Triangular Learning Rates

Slanted Triangular Learning Rates

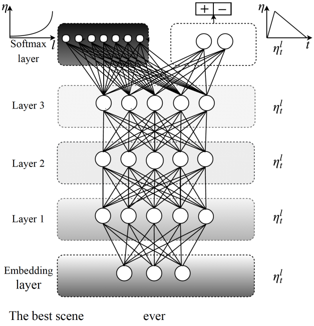


Target Task Classifier Fine-tuning

Target Task Classifier Fine-tuning



Target Task Classifier Fine-tuning



Classifier fine-tuning

- Two additional linear blocks;
- Concat pooling & Gradual unfreezing;
- BPTT for Text Classification (BTT3C);
- Bidirectional LM;

Target task classifier fine-tuning

Target task classifier fine-tuning

Concat Pooling

$$\mathbf{h}_c = [\mathbf{h}_T, \mathsf{maxpool}(\mathbf{H}), \mathsf{meanpool}(\mathbf{H})]$$
$$\mathbf{H} = \{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$$

Target task classifier fine-tuning

Concat Pooling

$$\mathbf{h}_c = [\mathbf{h}_T, \text{maxpool}(\mathbf{H}), \text{meanpool}(\mathbf{H})]$$
$$\mathbf{H} = \{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$$

Gradual Unfreezing

- Unfreeze layers one by one from the last;

Target task classifier fine-tuning

Concat Pooling

$$\mathbf{h}_c = [\mathbf{h}_T, \text{maxpool}(\mathbf{H}), \text{meanpool}(\mathbf{H})]$$
$$\mathbf{H} = \{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$$

Gradual Unfreezing

- Unfreeze layers one by one from the last;

BPT3C

- Fixed length batches for LM fine-tuning;

Target task classifier fine-tuning

Concat Pooling

$$\mathbf{h}_c = [\mathbf{h}_T, \text{maxpool}(\mathbf{H}), \text{meanpool}(\mathbf{H})]$$
$$\mathbf{H} = \{\mathbf{h}_1, \ldots, \mathbf{h}_T\}$$

Gradual Unfreezing

- Unfreeze layers one by one from the last;

BPT3C

- Fixed length batches for LM fine-tuning;

Bidirectional LM

- Pretrain both forward and backward LM;
- Average prediction for fine-tuned classifier;

Experimental Setup

Experimental Setup

Tasks and Datasets

- Sentiment Analysis: IMDb & Yelp reviews;
- Question Classification: TREC;
- Topic Classification: AG news & DBpedia ontology;

Experimental Setup

Tasks and Datasets

- Sentiment Analysis: IMDb & Yelp reviews;
- Question Classification: TREC;
- Topic Classification: AG news & DBpedia ontology;

Setup

- Preprocessing: tokens for upper-case, elongation, and repetition;
- Hyper-parameters: AWD-LSTM LM [Merity et al. 2017];

Experimental Setup

Tasks and Datasets

- Sentiment Analysis: IMDb & Yelp reviews;
- Question Classification: TREC;
- Topic Classification: AG news & DBpedia ontology;

Setup

- Preprocessing: tokens for upper-case, elongation, and repetition;
- Hyper-parameters: AWD-LSTM LM [Merity et al. 2017];

Baselines

- IMDb & TREC-6: CoVe [Mccann et al. 2017];
- AG, Yelp, DBpedia: [Johnson et al. 2017];

Results

Results

	Model	Test		Model	Test
IMDb	CoVe (McCann et al., 2017)	8.2	TREC-6	CoVe (McCann et al., 2017)	4.2
	oh-LSTM (Johnson and Zhang, 2016)	5.9		TBCNN (Mou et al., 2015)	4.0
	Virtual (Miyato et al., 2016)	5.9		LSTM-CNN (Zhou et al., 2016)	3.9
	ULMFiT (ours)	**4.6**		ULMFiT (ours)	**3.6**

Table 2: Test error rates (%) on two text classification datasets used by McCann et al. (2017).

	AG	DBpedia	Yelp-bi	Yelp-full
Char-level CNN (Zhang et al., 2015)	9.51	1.55	4.88	37.95
CNN (Johnson and Zhang, 2016)	6.57	0.84	2.90	32.39
DPCNN (Johnson and Zhang, 2017)	6.87	0.88	2.64	30.58
ULMFiT (ours)	**5.01**	**0.80**	**2.16**	**29.98**

Table 3: Test error rates (%) on text classification datasets used by Johnson and Zhang (2017).

Overview

Low-shot learning & Pretraining

Low-shot learning & Pretraining

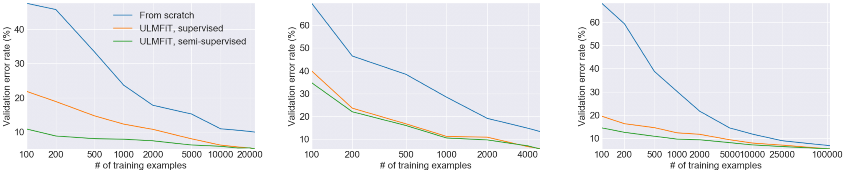


Figure 3: Validation error rates for supervised and semi-supervised ULMFiT vs. training from scratch with different numbers of training examples on IMDb, TREC-6, and AG (from left to right).

Low-shot learning & Pretraining

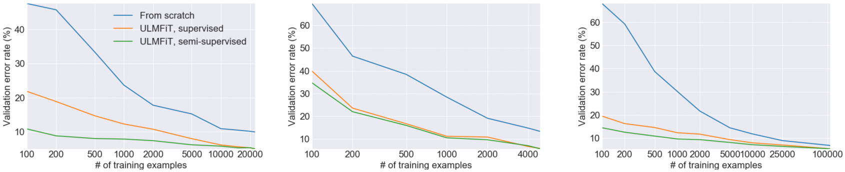


Figure 3: Validation error rates for supervised and semi-supervised ULMFiT vs. training from scratch with different numbers of training examples on IMDb, TREC-6, and AG (from left to right).

Pretraining	IMDb	TREC-6	AG
Without pretraining	5.63	10.67	5.52
With pretraining	**5.00**	**5.69**	**5.38**

Table 4: Validation error rates for ULMFiT with and without pretraining.

LM Quality & Fine-tuning

LM Quality & Fine-tuning

LM	IMDb	TREC-6	AG
Vanilla LM	5.98	7.41	5.76
AWD-LSTM LM	**5.00**	**5.69**	**5.38**

Table 5: Validation error rates for ULMFiT with a vanilla LM and the AWD-LSTM LM.

LM Quality & Fine-tuning

LM	IMDb	TREC-6	AG
Vanilla LM	5.98	7.41	5.76
AWD-LSTM LM	**5.00**	**5.69**	**5.38**

Table 5: Validation error rates for ULMFiT with a vanilla LM and the AWD-LSTM LM.

LM fine-tuning	IMDb	TREC-6	AG
No LM fine-tuning	6.99	6.38	6.09
Full	5.86	6.54	5.61
Full + discr	5.55	6.36	5.47
Full + discr + stlr	**5.00**	**5.69**	**5.38**

Table 6: Validation error rates for ULMFiT with different variations of LM fine-tuning.

Classifier Fine-tuning

Classifier Fine-tuning

Classifier fine-tuning	IMDb	TREC-6	AG
From scratch	9.93	13.36	6.81
Full	6.87	6.86	5.81
Full + discr	5.57	6.21	5.62
Last	6.49	16.09	8.38
Chain-thaw	5.39	6.71	5.90
Freez	6.37	6.86	5.81
Freez + discr	5.39	5.86	6.04
Freez + stlr	5.04	6.02	5.35
Freez + cos	5.70	6.38	**5.29**
Freez + discr + stlr	**5.00**	**5.69**	5.38

Table 7: Validation error rates for ULMFiT with different methods to fine-tune the classifier.

Classifier Fine-tuning Behavior

Classifier Fine-tuning Behavior

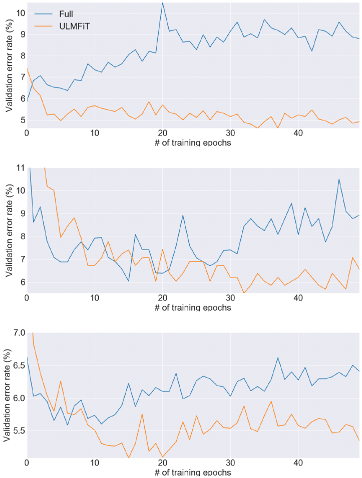


Figure 4: Validation error rate curves for fine-tuning the classifier with ULMFiT and '*Full*' on IMDb, TREC-6, and AG (top to bottom).

Overview

Thank You!
Questions & Comments?

References

Yosinski Jason, Clune Jeff, Bengio Yoshua, Lipson Hod (2014)
How transferable are features in deep neural networks?
NIPS 3320–3328.

Long Mingsheng, Cao Yue, Wang Jianmin, Jordan Michael I (2015)
Learning Transferable Features with Deep Adaptation Networks.
ICML volume 37.

Sharif Razavian Ali, Azizpour Hossein, Sullivan Josephine, Carlsson Stefan (2014)
CNN features off-the-shelf: an astounding baseline for recognition.
CVPR 806–813.

Hariharan Bharath, Arbeláez Pablo, Girshick Ross, Malik Jitendra (2015)
Hypercolumns for object segmentation and fine-grained localization.
CVPR 447–456.

Peters Matthew E, Ammar Waleed, Bhagavatula Chandra, Power Russell (2017)
Semi-supervised sequence tagging with bidirectional language models.
ACL

References

Rei Marek (2017)
Semi-supervised multitask learning for sequence labeling.
ACL

Liu Liyuan, Shang Jingbo, Xu Frank, Ren Xiang, Gui Huan, Peng Jian, Han Jiawei (2018)
Empower sequence labeling with task-aware neural language model.
AAAI

Dai Andrew M, Le Quoc V (2015)
Semi-supervised sequence learning.
NIPS 3079–3087

Merity Stephen, Keskar Nitish Shirish, Socher Richard (2017)
Regularizing and optimizing LSTM language models.
arXiv preprint arXiv:1708.02182

References

McCann Bryan, Bradbury James, Xiong Caiming, Socher Richard (2017)
Learned in translation: Contextualized word vectors.
NIPS 6294–6305

Johnson Rie, Zhang Tong (2017)
Deep pyramid convolutional neural networks for text categorization.
ACL 562–570