SEMI-SUPERVISED LEARNING FOR TEXT CLASSIFICATION BY LAYER PARTITIONING

Alexander Hanbo Li, Abhinav Sethy

Alexa AI, Amazon

ABSTRACT

Most recent neural semi-supervised learning algorithms rely on adding small perturbation to either the input vectors or their representations. These methods have been successful on computer vision tasks as the images form a continuous manifold, but are not appropriate for discrete input such as sentence. To adapt these methods to text input, we propose to decompose a neural network M into two components F and U so that $M = U \circ F$. [1] The layers in F are then frozen and only the layers in U will be updated during most time of the training. In this way, F serves as a feature extractor that maps the input to high-level representation and adds systematical noise using dropout. We can then train U using any state-of-the-art SSL algorithms such as Π-model, temporal ensembling, mean teacher, etc. Furthermore, this gradually unfreezing schedule also prevents a pretrained model from catastrophic forgetting. The experimental results demonstrate that our approach provides improvements when compared to state of the art methods especially on short texts.

***Index Terms*—** Semi-supervised learning, transfer learning, text classification, neural network

1. INTRODUCTION

Semi-supervised learning (SSL) [1] has been proved powerful for leveraging unlabeled data when we lack the resources to create large scale labeled dataset. In contrast with supervised learning methods that could only use labeled examples, SSL effectively uses the unlabeled samples to learn the underlying distribution of the data. Most SSL algorithms rely on an extra consistency or smoothness regularization which enforces the model to make consistent predictions on an input and its slightly perturbed version [2, 3, 4, 5, 6, 7]. The perturbation, however, is always made by adding artificial noise (e.g. Gaussian noise) to the input x. In image classification tasks, the inputs are images that can be represented by dense vectors in a continuous space. However, in text classification tasks, each input text is a sequence of tokens and each token is represented by an one-hot vector which forms a sparse high-dimensional space. Even when we use word embeddings, the underlying input sentences themselves are still discrete. In this case, adding continuous noises to the discrete inputs become inappropriate. To tackle this problem, [8] proposed to perturb each word by adding adversarial noise to the word embedding. However, it is hard to understand how independently adding noise to each token changes the sentence, as the perturbed embedding does not map back to any word. Ideally, after applying a perturbation function to a sentence, the noisy output should still represent a proper sentence and carries similar meaning.

Large-scale pretraining and transfer learning has achieved huge success on many NLP tasks [9, 10, 11, 12, 13, 14, 15, 16, 17]. The models are pretrained on some large dataset like Wikipedia's pages and then adapted to new domains by fine-tuning. For example, ULMFiT [15] divides the training procedure for text classification into three steps: (1) general domain language model (LM) pretraining, (2) target task LM fine-tuning and (3) target task classifier fine-tuning. Following these steps, they obtained state-of-the-art results on various text classification tasks. However, the fine-tuning step (2) seems redundant especially when the new-domain texts are short, and hence we ask ourselves whether the same level of performance can be achieved without LM fine-tuning.

To tackle the previous question, we propose one framework that allows adapting successful techniques in computer vision to NLP tasks. We avoid using artificial noises by decomposing a neural network into two components F and U so that $M = U \circ F$, and freeze F as both feature extractor and perturbation function. Because of the general-domain pretraining, layers in F carry domain-agnostic knowledge while layers in U are more task-specific. By training U on the outputs of F, we can use any state-of-the-art SSL algorithm that depends on continuous input variations. The similar technique is also used in ULMFiT [15] to effectively prevent catastrophic forgetting. In the experiments, we combine the proposed layer partitioning (**LayerParti**) with Π-model or temporal ensembling (TE) [3], and test their performance to IMDB [18] sentimental analysis and TREC-6 [19] text classification. LayerParti achieves comparable result on IMDB and better result on TREC-6 compared with ULMFiT. Finally, we also apply our method to a spoken personal memory retrieval task and achieve better performance using only 6% of the labels.

The paper is organized as follows. In Section 2, we review some most relevant work. In Section 3, we describe our method of layer partitioning for semi-supervised learning. In

[1] "$\circ$" stands for composition, hence $U \circ F(x) = U(F(x))$.

Section 4, we evaluate the proposed method and discuss the results. In Section 5, we conclude by discussing our overall findings and contributions.

2. RELATED WORK

In NLP, models pretrained on large-scale dataset [14, 16, 17] learn useful general knowledge. One can hence transfer a pretrained model to new domains by fine-tuning it on domain-specific data. ULMFiT [15] standardizes the procedure into three steps: general domain language model (LM) pretraining, target task LM fine-tuning and target task classifier fine-tuning. It also gradually unfreezes the layers from top to bottom in order to prevent catastrophic forgetting. This idea is also used in computer vision, where one only fine-tunes the last layers [20, 21] of a pretrained model. The common strategy is to use the lower layers to extract general features of the input.

For a comprehensive review of SSL methods, we refer readers to [22, 23] or [24]. We summarize four most relevant SSL methods below for continuous inputs. Our proposed method can be combined with any of them and applied to text classification.

Π-Model The input and the prediction function in Π-Model are both stochastic, and hence it can produce different outputs for the same input x. Π-Model [3, 4] adds a consistency loss which encourages the distance between a network's output for different passes of x through the network to be small. However, the teacher network targets are very unstable and can change rapidly along the training.

Temporal Ensembling and Mean Teacher [3, 5] proposed to obtain a more stable target by setting the teacher network as an exponential moving average of model parameters from previous training steps, or by directly calculating the moving average of previous targets.

Virtual Adversarial Training Virtual Adversarial Training [6, 8] approximates an adversarial noise to add to the input so that prediction function will be mostly affected. Note that in NLP, the noise is added to word embedding.

3. SEMI-SUPERVISED LEARNING BY LAYER PARTITIONING

3.1. Partitioning Neural Network Layers

Let M be a neural network model with n layers, we could then split M into two parts U and F, where F contains the lower layers $\{1, \dots, l\}$ and U contains the higher layers $\{l+1, \dots, n\}$. This is demonstrated in Figure 1a. In a language model, the lower layers tend to learn more general knowledge [16, 15] and are domain-agnostic. Therefore, we propose to **freeze** the layers in F and use them as a feature extractor. We only update the task-specific layers in U. The similar strategy has been used in computer vision [20, 21]. In fact, assuming we have samples $x_1, \dots, x_N$, freezing F is equivalent to training the model U with transformed inputs $F(x_1), \dots, F(x_N)$.

Depending on the splitting level l, the feature mapping F may stand for the word embeddings or more abstract features containing complicated context information. In practice, we notice that freezing only the lowest layers gives better results than freezing intermediate or high layers. This is aligned with the intuition that higher layers learn task-specific features that cannot be directly transferred to new domains. In our experiments, we freeze the word embedding layer and the first layer of the transformer encoder.

3.2. Perturbation of Textual Input

F can then be used to add systematical noise to the input by using dropout [25], as shown in Figure 1a. Each time x passes through F, the output will contain different perturbation. In another word, instead of adding artificial noise to x to get $\tilde{x} \leftarrow x + \epsilon$, we now have $\tilde{x} \leftarrow F(x)$. Because the layers in F are pretrained on a general domain, $F(x)$ is more likely to contain the same text information than $x + \epsilon$. For example, in sentimental analysis, changing one word *happy* to *sad* can totally change the sentiment of a sentence. This situation will rarely happen because of the language model property of F.

3.3. Consistency Constraints

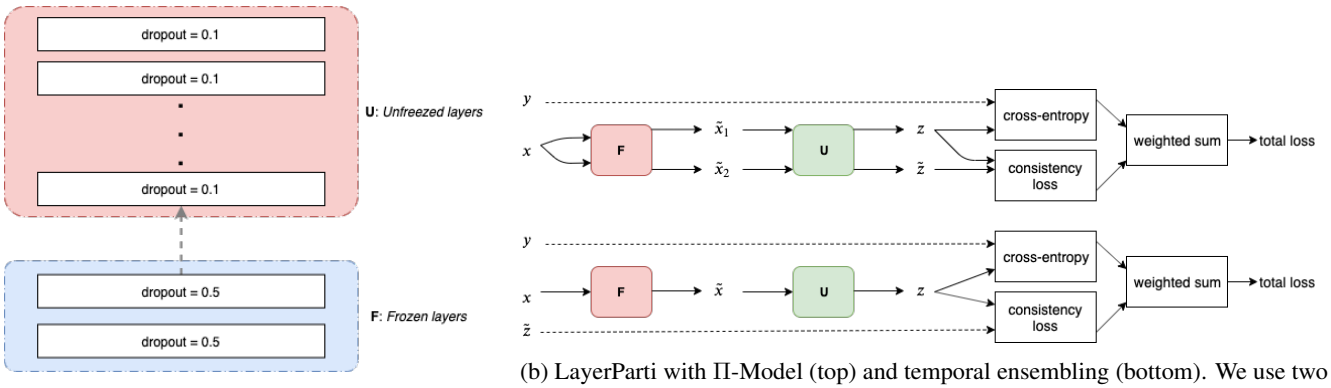
Now being able to properly perturb the discrete input, we can use any state-of-the-art semi-supervised learning method to train U. We choose two methods in this paper: Π-Model and temporal ensembling (TE) [3].

The diagrams of the two models are shown in Figure 1b. In Π-Model, each text x is passed through the frozen layers F twice to get two perturbed outputs $\tilde{x}_1$ and $\tilde{x}_2$, which are then fed into U to get two predictions z and $\tilde{z}$ of the class probabilities (i.e. $z_i \in \mathbb{R}^k$ where k is the number of classes). The final loss is then a weighted sum of the cross-entropy loss (CE) and the consistency loss. Following [3], we use mean squared error (MSE) as the consistency loss, and hence the total loss $L(x, y) = CE(z, y) + w(t) \cdot MSE(z, \tilde{z})$, where $w(t)$ is the weight of the consistency loss and is a function of the iteration t.

The temporal ensembling model is similar to Π-Model, except that the "teacher" targets $\tilde{z}$ is an ensemble of previous predictions. More formally, we update $Z_i \leftarrow \alpha Z_i + (1-\alpha) z_i$ and then set the target as $\tilde{z}_i = Z_i / (1 - \alpha^{T_i})$ where T_i is the number of times that x_i has been used. The factor $1 - \alpha^{T_i}$ is to correct the zero initialization bias [26, 3].

3.4. Gradually Unfreezing

Approaching the end of the training, we will gradually unfreeze the layers in F. The motivation is that U has been well trained on $\{F(x)\}$ and becomes saturated, we can then unfreeze F to



(a) Decomposition of a deep neural network.

(b) LayerParti with Π-Model (top) and temporal ensembling (bottom). We use two colors red and green to distinguish between *frozen* and *unfrozen* parts.

Fig. 1: The decomposition of a neural network and its application in SSL model training.

let it also learn some specific features of the task domain. We describe the algorithm in detail in Algorithm 1.

Algorithm 1 The training procedure of LayerParti.

Input: $\mathcal{D}_L$: labeled data, $\mathcal{D}_{UL}$: unlabeled data, $w(t)$: weight.
Initialize: Frozen layers F, unfrozen layers U. In this paper, F contains the word embedding and the first encoder layer. Set $z_i = 0$, $Z_i = 0$, $T_i = 0$ for all i.
while $t \leq$ max iterations **do**
  $B \leftarrow$ the new batch
  $\tilde{x}_i \leftarrow F(x_i)$, $z_i \leftarrow U(\tilde{x}_i)$ for all $i \in B$
  $L_{CE} \leftarrow \frac{1}{|\mathcal{D}_L \cap B|} \sum_{i \in \mathcal{D}_L \cap B} \text{CE}(z_i, y_i)$
  if using Π-Model **then**
    $\tilde{x}_i^{'} \leftarrow F(x_i)$, $\tilde{z}_i \leftarrow U(\tilde{x}_i^{'})$
  else if using Temporal Ensembling **then**
    $\tilde{z}_i \leftarrow Z_i / (1 - \alpha^{T_i})$
    $T_i \leftarrow T_i + 1$
    $Z_i \leftarrow \alpha Z_i + (1 - \alpha) z_i$
  end if
  $L_{consist} \leftarrow \frac{1}{|\mathcal{D}_{UL} \cap B|} \sum_{i \in \mathcal{D}_{UL} \cap B} MSE(z_i, \tilde{z}_i)$
  $L \leftarrow L_{CE} + w(t) L_{consist}$
  Back-propagate the gradients ∇L and update the layers in U.
  if $t \geq 80\%$ max iterations **then**
    If F is not empty, unfreeze the top layer in F and put it to U.
  end if
end while

4. EXPERIMENTS

We test the proposed method *LayerParti* with Π-Model or temporal ensembling (TE), but note that our method can be combined with other SSL algorithms like mean teacher [5], SWA [27, 28], FGE [29], etc.

4.1. Public Datasets

IMDB [30] dataset consists of movie reviews from the Internet Movie Database (IMDb) labeled as positive or negative. The TREC-6 dataset [19] consists of open-domain, fact-based questions divided into six semantic categories. We summarize the statistics of the datasets in Table 1. For both datasets, we truncate the texts to not exceed 256 tokens.

	Train	Test	Unlabeled	Avg	Max
IMDB	25,000	25,000	50,000	239	2,506
TREC-6	5,452	500	-	10	45

Table 1: Statistics of the datasets. "Train" and "Test" mean the number of labeled data in training and test partitions respectively. "Avg" and "Max" refer to the average and maximum text length.

4.2. Experimental Settings

We use a Transformer encoder from [31] with 16 layers, 410 dimensional embedding, 2100 dimensional hidden layer and 10 heads for each multi-head attention layer. The encoder was pretrained with a linear language model heading on the WikiText-103 [32] data. We also use the BERT-tokenizer from [31] where a [CLS] token is appended to each sentence. Then we simply add a linear classification layer on top of the embedding of the [CLS] token to predict the class.

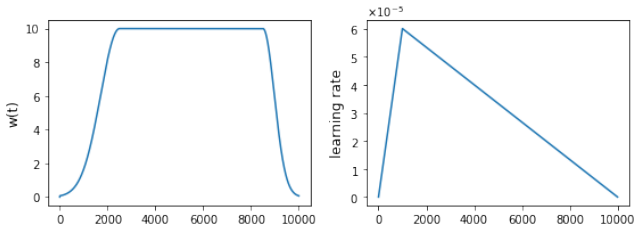


Fig. 2: The schedule of $w(t)$ with max value 10. We use the same scheduling function as in [3].

For supervised learning baselines, we train the model for 3

epochs with batch size 16 and we also clip the gradient norm to be less than 0.4. We choose Adam [26] optimizer with default parameter settings. The learning rate is triangular as shown in Figure 2 with warm-up proportion as 10%. The weight $w(t)$ has the same scheduling function as in [3] with 25% of the iterations for warm-up and 15% for ramp-down. With unlabeled data, we instead train each model for 8 epochs with 4 gradient accumulation steps. We sample the data so that 25% of each batch are labeled. The dropout rate in F is 0.5 for Π-Model and 0.3 for temporal ensembling, and the dropout rate in U is always 0.1.

4.3. Quantitative Results

From Table 2, LayerParti combined with Π-Model or TE provide the same level of results compared with SOTA [33]. On IMDB (long texts), our results are comparable to supervised ULMFiT results in [15] where they fine-tuned both the language model and the classifier on supervised data. But we do notice that their semi-supervised results (fine-tuning LM on all data and training classifier on labeled ones) are better than ours. However, on TREC-6, with all the labels available, our method acts as a regularization method and achieves 97.2% test accuracy that is even better than the best semi-supervised ULMFiT result of 96.4%. Also with only 60 labels, we can produce better accuracy than the semi-supervised ULMFiT results [15] with 100 labels.

The previous analysis suggests that on short texts, our method is very competitive and can give better accuracy. But on longer texts like IMDB reviews, fine-tuning the LM on all the data (not just on supervised ones) is indeed beneficial.

4.4. Personal Memory Retrieval

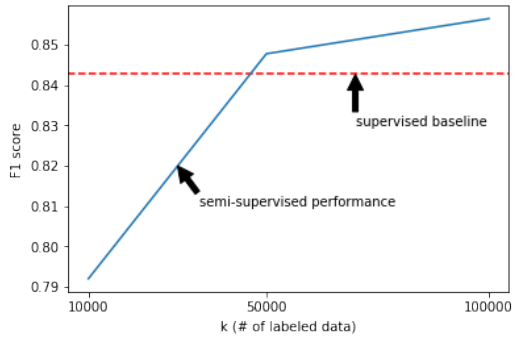


Fig. 3: Comparison of F1 scores. The *supervised baseline* is trained on all the examples. To get *semi-supervised* performance, we trained the models on k labeled data (half positive and half negative) and masked out the label information of the rest.

Recent progress of machine learning has largely enriched the functionality of smart personal assistants, including allowing users to store and retrieve long-term personal memory [34]. We apply our method to such a task where the memory is created by voice input and converted to text using a speech recognition system. The queries are also spoken and transcribed automatically.

On this task, we change the model to have two encoder layers, 300 dimensional embedding, 512 dimensional hidden layer and 5 heads. The full training dataset contains 871K labeled examples among which 52K are positive and 819K are negative. For supervised training, we use a sampler to guarantee each batch contains balanced positive and negative examples. The average text length is 7 and we truncate all the input at length 10. The test data contains 93K examples with 3K positives and 90K negatives. The results are in Figure 3. We observe that our method achieves better F1 using only 50,000 labeled data which is less than 6% of the labels used by the supervised baseline.

	Accuracy
IMDB(all)*	90.5%
IMDB(500)	81.5%
IMDB(100)	64.7%
IMDB(all+unsup) + LayerParti (Π)	91.4%
IMDB(500 + unsup) + LayerParti (Π)	83.2%
IMDB(100 + unsup) + LayerParti (Π)	69.3%
IMDB(all+unsup) + LayerParti (TE)	**91.8%**
IMDB(500 + unsup) + LayerParti (TE)	**85.1%**
IMDB(100 + unsup) + LayerParti (TE)	**75.9%**
TREC-6(all)	95.8%
TREC-6(400)	81.8%
TREC-6(60)	44.6%
TREC-6(all) + LayerParti (Π)	**97.2%**
TREC-6(400) + LayerParti (Π)	85.8%
TREC-6(60) + LayerParti (Π)	65.4%
TREC-6(all) + LayerParti (TE)	96.4%
TREC-6(400) + LayerParti (TE)	**86.0%**
TREC-6(60) + LayerParti (TE)	**67.8%**

Table 2: Accuracy results on IMDB and TREC-6 text classifications. *The IMDB baseline using all the data is worse than the 94.1% accuracy of VAT [6] but they are using 400 max length for text truncation while ours is 256.

5. CONCLUSION

We proposed a semi-supervised learning framework (LayerParti) for discrete text input. When combined with Π-Model or temporal ensembling – both proposed for image inputs – our method achieves competitive results on three text classification tasks. Especially when dealing with short texts like TREC-6 or personal memory retrieval, our framework provides better performance without LM fine-tuning.

6. REFERENCES

[1] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien, "Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]," *IEEE Transactions on Neural Networks*, vol. 20, no. 3, pp. 542–542, 2009.

[2] Philip Bachman, Ouais Alsharif, and Doina Precup, "Learning with pseudo-ensembles," in *Advances in Neural Information Processing Systems*, 2014, pp. 3365–3373.

[3] Samuli Laine and Timo Aila, "Temporal ensembling for semi-supervised learning," *arXiv preprint arXiv:1610.02242*, 2016.

[4] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen, "Regularization with stochastic transformations and perturbations for deep semi-supervised learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 1163–1171.

[5] Antti Tarvainen and Harri Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Advances in neural information processing systems*, 2017, pp. 1195–1204.

[6] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii, "Virtual adversarial training: a regularization method for supervised and semi-supervised learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1979–1993, 2018.

[7] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le, "Semi-supervised sequence modeling with cross-view training," *arXiv preprint arXiv:1809.08370*, 2018.

[8] Takeru Miyato, Andrew M Dai, and Ian Goodfellow, "Adversarial training methods for semi-supervised text classification," *arXiv preprint arXiv:1605.07725*, 2016.

[9] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[10] Andrew M Dai and Quoc V Le, "Semi-supervised sequence learning," in *Advances in neural information processing systems*, 2015, pp. 3079–3087.

[11] Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin, "How transferable are neural networks in nlp applications?," *arXiv preprint arXiv:1603.06111*, 2016.

[12] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher, "Learned in translation: Contextualized word vectors," in *Advances in Neural Information Processing Systems*, 2017, pp. 6294–6305.

[13] Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power, "Semi-supervised sequence tagging with bidirectional language models," *arXiv preprint arXiv:1705.00108*, 2017.

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

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

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

[17] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, "Improving language understanding by generative pre-training," *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

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

[19] Xin Li and Dan Roth, "Learning question classifiers," in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 2002, pp. 1–7.

[20] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International conference on machine learning*, 2014, pp. 647–655.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[22] Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the 20th International conference on Machine learning (ICML-03)*, 2003, pp. 912–919.

[23] Xiaojin Jerry Zhu, "Semi-supervised learning literature survey," Tech. Rep., University of Wisconsin-Madison Department of Computer Sciences, 2005.

[24] Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow, "Realistic evaluation of deep semi-supervised learning algorithms," in *Advances in Neural Information Processing Systems*, 2018, pp. 3235–3246.

[25] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[26] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson, "Averaging weights leads to wider optima and better generalization," *arXiv preprint arXiv:1803.05407*, 2018.

[28] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson, "There are many consistent explanations of unlabeled data: Why you should average," *arXiv preprint arXiv:1806.05594*, 2018.

[29] Timur Garipov, Pavel Izmailov, Dmitrii Podoprikhin, Dmitry P Vetrov, and Andrew G Wilson, "Loss surfaces, mode connectivity, and fast ensembling of dnns," in *Advances in Neural Information Processing Systems*, 2018, pp. 8789–8798.

[30] Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*. Association for Computational Linguistics, 2011, pp. 142–150.

[31] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew, "Transformers: State-of-the-art natural language processing," 2019.

[32] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher, "Pointer sentinel mixture models," *arXiv preprint arXiv:1609.07843*, 2016.

[33] "NLP-progress," `https://nlpprogress.com/`, Repository to track the progress in NLP, including the datasets and the current state-of-the-art for the most common NLP tasks.

[34] Rasool Fakoor, Amanjit Kainth, Siamak Shakeri, Christopher Winestock, Abdel-rahman Mohamed, and Ruhi Sarikaya, "Direct optimization of f-measure for retrieval-based personal question answering," in *2018 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2018, pp. 815–822.