Two-Stage Fine-Tuning for Improved Bias and Variance for Large Pretrained Language Models

Lijing Wang[1*] **Yingya Li**[2*] **Timothy Miller**[2] **Steven Bethard**[3] **Guergana Savova**[2]

[1]New Jersey Institute of Technology, lijing.wang@njit.edu

[2]Boston Children's Hospital and Harvard Medical School, {firstname.lastname}@childrens.harvard.edu

[3] University of Arizona, bethard@email.arizona.edu

Abstract

The bias-variance tradeoff is the idea that learning methods need to balance model complexity with data size to minimize both under-fitting and over-fitting. Recent empirical work and theoretical analyses with over-parameterized neural networks challenge the classic bias-variance trade-off notion suggesting that no such trade-off holds: as the width of the network grows, bias monotonically decreases while variance initially increases followed by a decrease. In this work, we first provide a variance decomposition-based justification criteria to examine whether large pretrained neural models in a fine-tuning setting are generalizable enough to have low bias and variance. We then perform theoretical and empirical analysis using ensemble methods explicitly designed to decrease variance due to optimization. This results in essentially a two-stage fine-tuning algorithm that first ratchets down bias and variance iteratively, and then uses a selected fixed-bias model to further reduce variance due to optimization by ensembling. We also analyze the nature of variance change with the ensemble size in low- and high-resource classes. Empirical results show that this two-stage method obtains strong results on SuperGLUE tasks and clinical information extraction tasks. Code and settings are available: https://github.com/christa60/bias-var-fine-tuning-plms.git

1 Introduction

Transformer-based neural language models, such as Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019), have achieved state-of-the-art (SOTA) performance for a variety of natural language processing (NLP) tasks through the process of *fine-tuning*. Given an NLP task, the process often involves searching for optimal pretrained models and hyperparameters while continuing to train the pretrained model on a domain-specific dataset, with the aim of building generalizable and robust fine-tuned models.

Based on the classic notion of the bias-variance tradeoff (Geman et al., 1992), where increasing model capacity decreases bias but increases variance (leading to a U-shaped test error curve), large pretrained models (LPMs) should have large variance and overfit domain-specific data which is often sparsely labeled and extremely imbalanced for classification. However, recent empirical work and theoretical analysis of neural networks challenge this classic bias-variance trade-off notion (Neal et al., 2018; Yang et al., 2020). It has been suggested that no such trade-off holds: as the width/depth of the network grows, bias monotonically decreases while variance initially increases followed by a decrease. This is why transformer-based LPMs often achieve better performance compared to less complex models like long short-term memory (LSTM)-based models or feature-rich methods (e.g., support vector machines). In the context of the new bias-variance paradigm, these LPMs are seemingly complex enough to have low bias and variance, however, so far there has been no method to justify whether those SOTA models are generalizable and robust in solving a variety of downstream tasks. In this paper, we (1) show that many SOTA models are very sensitive to data and training randomness, and (2) provide a variance decomposition-based justification method.

We also aim to improve model performance, reducing the generalization error of LPMs by reducing their bias and variance. Recent findings in Yang et al. (2020) show that the generalization error mainly comes from bias. Bias can be reduced by modifying the model architecture, e.g., making the neural networks wider and deeper as in transformer-based LPMs. However, pretraining new or larger language models can be challenging due to the technical and computational resource

[*] co-first authors

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 15746–15761
July 9-14, 2023

requirements afforded by only a few institutions – a topic outside the scope of this paper. We focus on the problem of reducing variance of neural models to further boost model performance, given a fixed bias (i.e., a fixed pretrained model). Ensemble methods have been successful in boosting predictive performance of single learners (Ren et al. (2016) presents a comprehensive review) and thus are a promising venue to explore.

We propose a two-stage fine-tuning framework that first justifies the generalization status of a selected pretrained model through a concrete metric, and then uses the fixed-bias model to further reduce variance due to optimization through ensembling techniques. To the best of our knowledge, we are the first to provide such a metric and perform theoretical and empirical analysis using ensembles for improved bias and variance for LPMs. We conduct experiments on the SuperGLUE tasks as well as on information extraction from clinical text as it is a domain of high significance and presents data challenges due to the limitations of sharing patient-related data. We believe our proposal is of interest to any unstructured domain where neural models are used. Specifically we make the following contributions:

- We propose a two-stage fine-tuning algorithm for improving bias and variance in the new bias-variance paradigm;
- We provide a variance decomposition-based strategy to examine whether LPMs in fine-tuning settings are generalizable enough to have low bias and variance;
- We perform theoretical and empirical analyses using ensembles explicitly designed to decrease variance due to optimization while keeping bias unchanged;
- We analyze the nature of variance change due to ensembling in low- and high-resource classes in classification tasks;
- We conduct comprehensive experiments and show that the proposed two-stage method obtains strong results on SuperGLUE tasks and two clinical NLP tasks.

2 Preliminaries

In this section we present the bias-variance decomposition for squared loss in the new paradigm studied in Neal et al. (2018) and Yang et al. (2020). We also present a further decomposition of variance. We denote f as a supervised learning task such that $f : \mathcal{X} \rightarrow \mathcal{Y}$, based on a training dataset S of m i.i.d. samples drawn from a joint distribution $\mathcal{D}$ of $(\mathcal{X}, \mathcal{Y})$. The learning target is to minimize the mean squared error $\mathcal{E}(f) = \mathbb{E}_{(x,y)}\left[\| y - f(x) \|^2\right]$, where $(x, y) \sim \mathcal{D}$.

We consider the predictor f_θ as a random variable depending on the random variable S for training dataset and the random variable O for optimization randomness, where $\theta = \mathcal{A}(S, O) \in \mathbb{R}^p$ represents the weights of neural networks produced by the learning algorithm $\mathcal{A}$. p denotes the dimension of θ. The notations and their descriptions are shown in Table 3 in the Appendix.

2.1 Bias-variance decomposition

In the context of classification of C classes, where $y \in \mathbb{R}^C$ is represented as a one-hot vector and $f_\theta(x) \in \mathbb{R}^C$ denotes an output probability vector by the predictor, the risk $\mathcal{R}$ of the learning algorithm can be decomposed into three sources of errors (Geman et al., 1992):

$$\mathcal{R} = \mathcal{E}_{noise} + \mathcal{E}_{bias} + \mathcal{E}_{var} \tag{1}$$

The first term is the irreducible noise and comes from the intrinsic error of data independent of the predictor. The second is a bias term:

$$\mathcal{E}_{bias} = \mathbb{E}_{(x,y)}\left[\| \mathbb{E}_\theta[f_\theta(x)] - \mathbb{E}[y|x] \|^2\right] \tag{2}$$

The third is a variance term

$$\begin{aligned} \mathcal{E}_{var} &= \mathbb{E}_x \mathrm{Var}(f_\theta(x)) \\ &= \mathbb{E}_x\left[\mathbb{E}_\theta\left[\| f_\theta(x) - \mathbb{E}_\theta[f_\theta(x)] \|^2\right]\right] \end{aligned} \tag{3}$$

and can be further decomposed into the *variance due to optimization* Var_{opt} and the *variance due to sampling* Var_{samp} (Neal et al., 2018):

$$\begin{aligned} \mathrm{Var}(f_\theta(x)) &= \mathrm{Var}_{opt} + \mathrm{Var}_{samp} \\ &= \mathbb{E}_S[\mathrm{Var}_O(f_\theta(x)|S)] \\ &+ \mathrm{Var}_S(\mathbb{E}_O[f_\theta(x)|S]) \end{aligned} \tag{4}$$

where we denote the expectation of the decomposed variance as $\mathcal{E}_{var_O} = \mathbb{E}_x[\mathrm{Var}_{opt}]$ and $\mathcal{E}_{var_S} = \mathbb{E}_x[\mathrm{Var}_{samp}]$.

2.2 Theoretical findings from variance decomposition

Assuming the learning task is to learn a linear mapping $y = \theta^T x + \epsilon$ where ϵ denotes the noise random

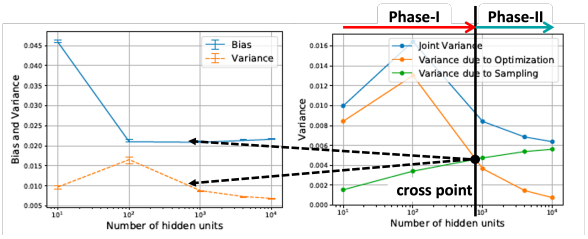


Figure 1: Empirical findings from Neal et al. (2018).

variable with $\mathbb{E}[\epsilon] = 0$ and $\text{Var}(\epsilon) = \sigma_\epsilon^2$, and the input vector is $x \in \mathbb{R}^p$ where p is the input or parameter dimensionality.

In this context, the *over-parameterized* setting is when $p > m$ and the *under-parameterized* setting is when $p \leq m$.

The theoretical findings in Neal et al. (2018) prove that $\mathcal{E}_{var}$ grows with p in the under-parameterized case, while in the over-parameterized case, the variance does not grow with p but scales with the dimension of the data:

$$\mathbb{E}_x \text{Var}(f_\theta(x)) = \begin{cases} \frac{p}{m}\sigma_\epsilon^2 & \text{for} \quad p \leq m \\ \frac{r}{m}\sigma_\epsilon^2 & \text{for} \quad p > m \end{cases} \tag{5}$$

where $r = rank(X)$ and $X \in \mathbb{R}^{m \times p}$ denotes the data matrix whose ith row is the training point x_i^T. Furthermore, $\mathcal{E}_{var_O}$ vanishes as p increases under the linear squared regression assumption and $\mathcal{E}_{var_S}$ depends on critical parameter dimensionality $d(p)$.

2.3 Empirical findings from variance decomposition

Finding-I: as shown in the left panel of Figure 1[1], $\mathcal{E}_{bias}$ decreases quickly and levels off once sufficiently over-parameterized, while $\mathcal{E}_{var}$ is unimodal contrary to the classic theory. *Finding-II*: in the right panel of Figure 1, $\mathcal{E}_{var_O}$ is significant and higher than $\mathcal{E}_{var_S}$ in the under-parameterized regime. The two variances cross at a certain p once sufficiently over-parameterized. However, empirical p and m of the over-parameterized setting is not strictly following the theoretical findings in section 2.2. *Finding-III*: in multi-layer models where p is the width and q is the depth, given a fixed p, $\mathcal{E}_{bias}$ decreases while $\mathcal{E}_{var}$ increases as q increases (Yang et al., 2020). These empirical findings hold for multiple datasets in the original papers.

[1]The two underlying subplots are from Netzer et al. (2011) on the SVHN dataset

3 Two-Stage Fine-Tuning for Improved Bias and Variance

3.1 Overview

The prevailing fine-tuning methods first build or select an LPM and then fine-tune its parameters on the downstream datasets. The SOTA setting for the LPM and its best hyperparameters for fine-tuning are chosen based on evaluation results, such as precision (P), recall (R), F1 and accuracy scores, using grid-search or random-search. Given a fine-tuning task with a fixed training dataset, there is an upper limit to the learning ability of an LPM which is hard to measure by traditional evaluation methods. For a selected LPM, it is usually hard to decide when to stop searching for hyperparameters. Different from the prevailing fine-tuning setting, we propose a two-stage fine-tuning method. We first provide a variance decomposition-based justification method to roughly measure the generalization ability of a pretrained model w.r.t. the upper limit of its learning ability. In Stage-I, the SOTA setting is built by ratcheting down bias and variance in an iterative way. The searching loop stops until an acceptable performance appears or no more improvement is observed. In Stage-II, given the SOTA setting built in Stage-I, the variance due to optimization is reduced by ensembling techniques. Algorithm 1 outlines the procedure of the proposed two-stage fine-tuning method. The details of each stage are presented below.

3.2 Stage-I: Justification of generalization ability

Based on the preliminaries in Section 2.3, assuming an algorithm $\mathcal{A}(S, O)$ is fixed, the $\mathcal{E}_{bias}$, $\mathcal{E}_{var}$, $\mathcal{E}_{var_O}$, and $\mathcal{E}_{var_S}$ changes as p, q, and m change. Taking the crossing point ($\mathcal{E}_{var_O} = \mathcal{E}_{var_S}$) as a dividing point, we define the generalization ability $\mathbb{G}_p$ as:

$$\mathbb{G}_p = \begin{cases} \text{Phase-I} & \text{for} \quad \mathcal{E}_{var_O} > \mathcal{E}_{var_S} \\ \text{Phase-II} & \text{for} \quad \mathcal{E}_{var_O} \leq \mathcal{E}_{var_S} \end{cases} \tag{6}$$

where Phase-I implies large bias and variance leading to large generalization error. Phase-II implies small bias and variance leading to small generalization error which may be good enough w.r.t. the upper limit of the learning ability of $\mathcal{A}$.

Justification criteria: After each evaluation, if $\mathbb{G}_p$ is in Phase-I, it is necessary to explore more hyperparameter settings or new pretrained models until $\mathbb{G}_p$

is in Phase-II or an acceptable performance (e.g., P, R, F1) is achieved given the limited computing resources available in practice. Then fine-tuning can move to Stage-II. If $\mathbb{G}_p$ is in Phase-II, the current setting may be generalizable enough for the given learning task so that the searching can be stopped. Stage-II can be applied but is not necessary. We note that similar to *Finding-II* in Section 2.3, $\mathbb{G}_p$ cannot be determined directly based on p, q, and m. This breakdown provides a high-level guideline for evaluating the generalization of LPMs in an empirical way.

3.3 Stage-II: Ensembling to reduce variance

Ensembles have been proven to be able to improve prediction accuracy and consistency of any learning models in Bonab and Can (2019); Wang et al. (2020). Bagging-based ensembles which are commonly used in various learning tasks have been proven to be able to reduce $\mathcal{E}_{var}$ while keeping $\mathcal{E}_{bias}$ unchanged. However, no theoretical analysis has been discussed in the context of the variance decomposition paradigm. In Stage-II, we focus on bagging-based ensembles to further improve the model performance by reducing $\mathcal{E}_{var_O}$ while keeping $\mathcal{E}_{var_S}$ unchanged. Applying Stage-II can either move a model from Phase-I to Phase-II though ensembling, i.e., reducing $\mathcal{E}_{var_O}$ until $\mathcal{E}_{var_O} \leq \mathcal{E}_{var_S}$; or further improve a model's generalization ability from Phase-II by reducing $\mathcal{E}_{var_O}$.

We perform empirical analysis in Section 4 and theoretical analysis in Section 5 to investigate why and how bagging-based ensembles can guarantee such improvements in this context. We also analyze the nature of variance change with the ensemble size in low- and high-resource classes in classification tasks. Boosting ensembles have a more complex behaviour thus are out of scope for this paper.

Algorithm 1: Pseudocode of the two-stage fine-tuning method.

Input: S: training dataset; $\mathcal{A}$: optimization algorithm; N: number of ensemble single learners; $\mathcal{F}$: majority voting for ensemble;
Output: ζ: ensemble learner

```
/* Stage-I                                                    */
```
$\mathbb{G}_p \leftarrow$ Phase-I
$\mathcal{E}^* \leftarrow 0$
while *($\mathbb{G}_p$ = Phase-I) or ($\mathcal{E}^*$ is not acceptable)* **do**
  Choose a pretrained model f and an initialization seed O.
  $\theta \leftarrow \mathcal{A}(S, O, f)$
  Compute $\mathcal{E}_{var_O}$ and $\mathcal{E}_{var_S}$ by Equation 4.
  $\mathbb{G}_p \leftarrow$ Phase-I if $\mathcal{E}_{var_O} > \mathcal{E}_{var_S}$ otherwise Phase-II
  $\mathcal{E}^* \leftarrow$ score(f_θ) `// P,R,or F1.`
$\zeta := f$
```
/* Stage-II                                                   */
```
if *($\mathbb{G}_p \neq$ Phase-II) or ($\mathcal{E}^*$ is not satisfied)* **then**
  $\xi \leftarrow \emptyset$ `// The set of ensemble components.`
  N is set to be $\geq \mathcal{E}_{var_O}/\mathcal{E}_{var_S}$
  while *len(ξ)* < N **do**
    Resample training and validation sets S_i from S.
    Train f^* on S_i using snapshot learning and save N trained learners $f^*_{\theta_1}, \ldots, f^*_{\theta_N}$ where $\theta_i = \mathcal{A}(S_i, O^*, f^*)$.
    Select l trained learners from the saved ones by applying pruning algorithm (Wang et al., 2020).
    $\xi \cup f^*_{\theta_1}, \ldots, f^*_{\theta_l}$
  $\zeta := \mathcal{F}(\xi)$

4 Experiments

4.1 Data and models

We conduct experiments on the SuperGLUE tasks and two major clinical information extraction datasets. The data processing and statistics and hyperparameter settings are shown in Appendix Table 4 and Table 5. Their brief descriptions are:

- **SuperGLUE** (Wang et al., 2019) is a benchmark dataset designed for a more rigorous test of general-purpose language understanding systems after GLUE (Wang et al., 2018). We replicate the scores on dev set[2], and select six tasks (BoolQ, CB, RTE, MultiRC, WiC, and COPA). The selected tasks cover question answering (QA), natural language inference, and word sense disambiguation. The SOTA setting is based on the setting in Liu et al. (2019) using `roberta-large` as the pretrained model.

- **THYME** (Temporal Histories of Your Medical Events) corpus (Styler IV et al., 2014) for temporal relation extraction, consisting of 594 de-identified clinical and pathology notes on colon cancer patients. We use the THYME+ version of the corpus (Wright-Bettner et al., 2020). There are 10 classes of extremely imbalanced class distribution. The SOTA setting is based on the setting in Lin et al. (2021) using `PubmedBERTbase-MimicBig-EntityBERT` as the pretrained model.

[2] https://github.com/pytorch/fairseq/tree/master/examples/roberta

- **2012 i2b2 Temporal Relations** (Sun et al., 2013) consists of 394 training de-identified reports, 477 test de-identified reports, and 877 unannotated reports. There are 3 classes of slightly imbalanced class distribution. The SOTA setting is based on the setting in Haq et al. (2021) using `BioBERT-base-cased-v1.1` as the pretrained model.

4.2 Metrics and settings

We use an NVIDIA Titan RTX GPU cluster of 7 nodes for fine-tuning experiments through HuggingFace's Transformer API (Wolf et al., 2020) version 4.13.0. We leverage the `run_glue.py` pytorch version as our fine-tuning script. Unless specified, default settings are used in our experiments. Due to differences in the fine-tuning script and some missing settings not provided by the original authors, we were unable to reproduce the exact SOTA scores but we achieved scores close to the SOTA ones. Our implementation are denoted as replicated-SOTA (RSOTA). We compare our implementation and reference SOTA scores in Appendix Table 7. We use the RSOTA settings as the starting point to conduct the experiments. We use $\mathcal{E}_{\mathbf{bias}}$, $\mathcal{E}_{\mathbf{var}}$, $\mathcal{E}_{\mathbf{var_O}}$, and $\mathcal{E}_{\mathbf{var_S}}$ for Stage-I, and adopt classic evaluation metrics (**P**, **R**, and **F1**) for Stage-I and Stage-II. For the purposes of consistency, we report P, R, and F1 of SuperGLUE tasks for consistency with the reported results for the THYME and i2b2 tasks in the experiment results. Accuracy scores are reported in Appendix Table 7.

4.3 Experimental design

There are two stages in the proposed method. For Stage-I, we use 5 random seeds for the randomness over data samples S and 5 for the randomness over initialization O, resulting in a total of 25 fine-tuned models. The averages over data samples are performed by taking the training set S and creating 5 bootstrap replicate training/validation splits with the same class distribution. The bias expectation in Equation 2 is estimated as the averages over both S and O. The variance decomposition is estimated based on Equation 4. More specifically, $\mathcal{E}_{var_O}$ is estimated as the averages over S of the variance over O, and $\mathcal{E}_{var_S}$ is estimated as the variance over S of the averages over O. Furthermore, we also apply `RoBERTa-base-uncased (RBU)` as the pretrained model for each fine-tuning task using the RSOTA setting except for pretrained models. Their descriptions are shown in Appendix Table 6. To replicate SOTA scores and obtain RSOTA settings for each task, we conduct hyperparameter searching in an iterative way. This process is considered as the experiment of Stage-I.

	$\mathcal{E}_{\mathbf{bias}}$	$\mathcal{E}_{\mathbf{var}}$	$\mathcal{E}_{\mathbf{var_O}}$	$\mathcal{E}_{\mathbf{var_S}}$	$\mathbb{G}_{\mathbf{p}}$	F1
BoolQ-RBU	162	5.4	3.9	1.6	Phase-I	77.8
BoolQ-RSOTA	142	9.9	6.2	3.7	Phase-I	84.3
CB-RBU	175	0.2	0.1	0.1	-	49.2
CB-RSOTA	149	1.7	1.5	0.2	Phase-I	62.0
RTE-RBU	176	11.4	8.0	3.4	Phase-I	74.0
RTE-RSOTA	153	13.2	11.2	2.1	Phase-I	83.5
MultiRC-RBU	164	5.9	4.6	1.3	Phase-I	78.5
MultiRC-RSOTA	178	13.3	10.5	2.8	Phase-I	74.7
WiC-RBU	212	5.5	4.2	1.3	Phase-I	63.6
WiC-RSOTA	199	12.7	10.1	2.5	Phase-I	70.3
COPA-RBU	250	0.0	0.0	0.0	-	38.0
COPA-RSOTA	185	4.3	3.9	0.5	Phase-I	81.2
THYME-RBU	81	0.17	0.14	0.02	Phase-I	57.0
THYME-RSOTA	80	0.09	0.07	0.02	Phase-I	61.8
i2b2-RBU	150	0.76	0.62	0.14	Phase-I	76.8
i2b2-RSOTA	152	0.73	0.58	0.14	Phase-I	78.1

Table 1: Bias and variance of different pretrained models on the SuperGLUE, THYME and i2b2 datasets. RoBERTa-base-uncased (RBU). Values of $\mathcal{E}_{\mathbf{bias}}$, $\mathcal{E}_{\mathbf{var}}$, $\mathcal{E}_{\mathbf{var_O}}$, and $\mathcal{E}_{\mathbf{var_S}}$ are relative values to 0.001. F1 scores are the means over 5 random seeds for initialization.

For Stage-II, any bagging-based ensemble algorithms are feasible. In our preliminary experiments (Wang et al., 2022), we have shown that the dynamic snapshot ensemble algorithm (Wang et al., 2020), which we call ENS in this paper, works better than vanilla bagging ensembles. ENS is a bagging-based ensemble explicitly designed to reduce variance over optimization-related hyperparameters in one framework, with the aim of building computationally efficient strategies to boost model performance on top of any given setting with a guarantee (i.e., simple bagging ensemble cannot guarantee an improvement). In our implementation, we employ ENS. The ensemble size is 5 and majority voting is used to generate ensemble predictions. To explore the ensemble impact on low- and high-resource classes, we compute and compare performance improvements of each class from the extremely imbalanced THYME dataset. To investigate the impact of ensemble size on improving the model performance of imbalanced classes, we also evaluate performance of individual classes using ENS of size 1 to 10. We compute 95% confidence intervals for these estimates using bootstrapping over 5 samples. More details are in Appendix B.

	BoolQ			CB			RTE			MultiRC		
Method	P	R	F1	P	R	F1	P	R	F1	P	R	F1
RSOTA	84.58	84.10	84.34	60.76	63.80	62.06	83.94	83.40	83.54	73.02	77.54	74.68
	(±0.34)	(±0.32)	(±0.30)	(±19.0)	(±15.0)	(±16.9)	(±0.89)	(±0.94)	(±0.94)	(±21.8)	(±13.5)	(±18.76)
ENS	84.96	84.38	84.74	92.68	93.14	92.66	86.00	85.38	85.48	82.14	82.30	82.16
	(±0.34)	(±0.58)	(±0.47)	(±1.3)	(±2.8)	(±1.2)	(±0.79)	(±1.16)	(±1.10)	(±1.41)	(±1.41)	(±1.3)
IPV	0.45%	0.33%	0.47%	52.53%	45.99%	49.31%	2.45%	2.37%	2.32%	12.49%	6.14%	10.02%

	WiC			COPA			THYME			i2b2		
Method	P	R	F1	P	R	F1	P	R	F1	P	R	F1
RSOTA	72.06	70.74	70.30	82.54	81.72	81.24	66.6	58.2	61.8	78.3	76.9	78.1
	(±0.12)	(±1.7)	(±1.9)	(±11.40)	(±12.10)	(±12.4)	(±1.02)	(±1.47)	(±1.25)	(±1.56)	(±0.98)	(±1.24)
ENS	72.18	71.30	70.98	93.84	93.80	93.60	72.9	60.1	65.9	80.5	78.3	79.3
	(±0.17)	(±0.36)	(±0.42)	(±0.43)	(±0.48)	(±0.48)	(±0.86)	(±1.16)	(±0.95)	(±1.23)	(±0.79)	(±0.97)
IPV	0.17%	0.79%	0.97%	13.69%	14.78%	15.21%	9.46%	3.26%	6.63%	2.81%	1.82%	1.54%

Table 2: Ensemble model performance. Test set results with average and 95% confidence interval of 5 random samples, where the t value for 95% confidence is 2.776. ENS denotes ensemble of 5 components. IPV denotes improvement percentage by ENS compared with RSOTA.

4.4 Justification results

Table 1 shows $\mathcal{E}_{bias}$, $\mathcal{E}_{var}$, $\mathcal{E}_{var_O}$, and $\mathcal{E}_{var_S}$ computed on the datasets with different pretrained models. It is noted that in our experiment, we are not applying the algorithm for Stage-I to ratchet down bias and variance in an iterative manner. The goal of this table is to analyze both RBU and RSOTA models for the bias and variance trends discussed in Section 2. Interestingly, we observe that $\mathcal{E}_{var_O} > \mathcal{E}_{var_S}$ for all datasets and models except for CB-RBU and COPA-RBU where models are not well trained given that F1 score is around 0.5 indicating random guess. This implies that the vast majority of the SOTA models we experimented with are in Phase-I (i.e., not generalizable enough for their tasks), which is contrary to our intuition that these transformer-based models are complex enough given the moderate sized labeled datasets. It is also observable that $\mathcal{E}_{bias}$ is much larger than $\mathcal{E}_{var}$ indicating that the model performance is dominated more by bias than by variance. For SuperGLUE tasks, with the same hyperparameter setting (i.e., $\mathcal{A}(S, O)$), the RSOTA models (i.e., larger p and q than RBU models) achieve smaller $\mathcal{E}_{bias}$ but larger $\mathcal{E}_{var}$ than RBU models except for MultiRC. The change of $\mathcal{E}_{var}$ mainly comes from the change of $\mathcal{E}_{var_O}$. As *Finding-I* in Section 2.3 that $\mathcal{E}_{bias}$ decreases while $\mathcal{E}_{var}$ is unimodal, our observation implies that RBU models are before peak and long way toward Phase-II while RSOTA models get closer to Phase-II than RBU models. The exception is the result on the MultiRC dataset which is QA corpus listing a passage which consists of text, a question, and multiple answers to that question each with a true/false label. Although MultiRC represents a variety of question types (e.g. yes/no, factual, etc.), the passages are not annotated for question type. As explained in Appendix section B.1., we represented the QA pairs within each passage as `text, question, answer, label` instances and sampled from these instances. Using this instance-based sampling likely leads to samples not stratified by question types, therefore not necessarily representative. This probably explains the better mean F1 for MultiRC-RBU as compared to the mean F1 for MultiRC-RSOTA in Table 1 (different samples are created for each run). However, when we drill down to the best model F1 for MultiRC-RBU and MultiRC-RSOTA, the results are 78.9 and 85.4 F1-scores respectively, which supports the trend in Table 1.

For the SuperGLUE tasks, the bias and variance of RBU models and RSOTA models are shown together to illustrate a trend (like Fig. 1) as p and q are the only variables. However for THYME and i2b2 tasks, similar trends could not be interpreted since the RSOTA models are pretrained with domain specific corpora while the RBU models are pretrained with general corpora. This implies that for fine-tuning tasks such as temporal relation extraction, other factors (e.g., domain corpora used to pretrain models) may have larger impact than the model complexity. Our observations are consistent with *Finding-II* that empirical p and m setting is not strictly following theoretical findings which are under linear squared regression assumption. This also indicates that p, q, and m cannot be used to measure $\mathbb{G}_p$ empirically. On the other hand,

our variance decomposition-based method does not rely on p, q, and m, therefore it provides the basis for a more generalized justification method.

4.5 Ensemble results

Table 2 presents P, R, and F1 scores of RSOTA and ENS methods on all datasets. Similar to prior studies (e.g. Zhang et al., 2020; Du et al., 2022), results for the SuperGLUE tasks are reported on the dev set. The accuracy scores for each task are presented in the Appendix Table 7. Compared to the RSOTA setting, the ENS method boosts performance on all datasets, with the largest gains of 49.3% and 15.2% relative F1 improvements on the low-resource CB and COPA datasets respectively. In Section 5, we analyze why ensembles work from the variance decomposition perspective, which provides insights into how ensembles help reduce $\mathcal{E}_{var_O}$ and lead to better prediction accuracy.

4.6 Ensemble impact on low- and high-resource classes

We further investigate the improvements on low-resource datasets (e.g. CB and COPA). To eliminate all interference from p, q, $\mathcal{A}(S, O)$, pretrained models and only keep m as the variable, we tease apart the results of the extremely unbalanced THYME dataset and analyze the performance on each class. Its most frequent classes (i.e., *high-resource* classes) are CONTAINS (2895), OVERLAP (2359), and BEFORE (2241); and the least frequent classes (i.e., *low-resource* classes) are NOTED-ON (140), BEGINS-ON (160), and ENDS-ON (244). The initial F1 scores are: CONTAINS-0.776, OVERLAP-0.539, and BEFORE-0.469; NOTED-ON-0.618, BEGINS-ON-0.608, and ENDS-ON-0.695. In Figure 2, we show absolute improvement and improvement percentage of F1 with various ensemble size N (compared with single learners i.e., $N = 1$). These values are computed based on the mean with 95% confidence interval over 5 random samples for each class and each ensemble size. It is observable that given a fixed N, the performance improvements by F1 scores on the low-resource classes – NOTED-ON (brown), BEGINS-ON (red), and ENDS-ON (orange) – are larger than the ones of high-resource classes. The difference becomes larger as N increases. The scales of improvement are not affected by the initial results; i.e., the larger improvements on low-resource classes are not due to lower initial F1 scores. This is an interesting observation and may introduce a new solution for improving performance of imbalanced datasets. More similar results on P and R are shown in in Appendix Fig. 3. We explore theoretical insights into these observations in Section 5.

5 Discussion and Theoretical Analysis

5.1 Basic statistics

Let $X_1, X_2, \cdots, X_N$ be a random sample from a population with mean μ and variance σ^2 and $\bar{X} = \frac{1}{N}\sum_{i=1}^{N} X_i$. Then the following two items hold.

a: $\mathbb{E}[\bar{X}] = \mathbb{E}[X_i] = \mu$

b: $\text{Var}(\bar{X}) = \frac{1}{N}\text{Var}(X_i) = \frac{1}{N}\sigma^2$

5.2 Ensemble in bias-variance decomposition

We work in the context of bagging-based ensembles, assuming the ensemble predictor $\bar{f}(x)$ is $\bar{f}(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x)$ is the averaging of N single learners trained with different samples of S and O. Based on the basic statistics, the $\mathcal{E}_{bias}$ of $\bar{f}(x)$ in Equation 2 is unchanged while the $\mathcal{E}_{var}$ of $\bar{f}(x)$ in Equation 3 decreases by $\frac{1}{N}$. Furthermore, we have:

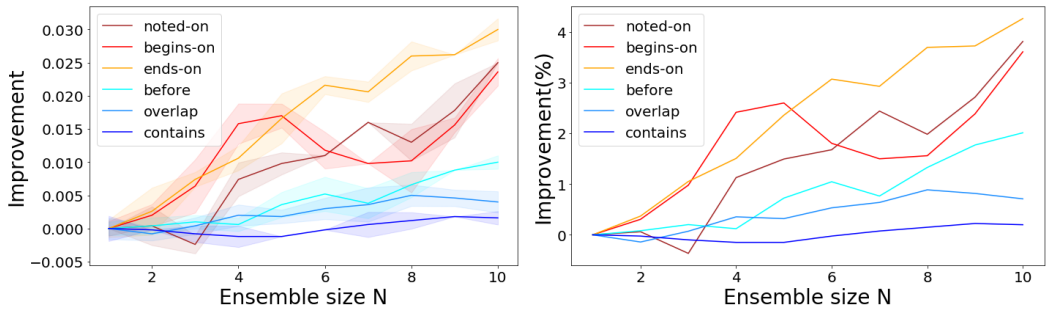
$$\begin{aligned}\mathcal{E}_{var_O} &= \mathbb{E}_x\left[\mathbb{E}_S[\text{Var}_O(\bar{f}_\theta(x)|S)]\right] \\ &= \frac{1}{N}\mathbb{E}_x\left[\mathbb{E}_S[\text{Var}_O(f_\theta(x)|S)]\right]\end{aligned} \tag{7}$$

and:

$$\begin{aligned}\mathcal{E}_{var_S} &= \mathbb{E}_x\left[\text{Var}_S(\mathbb{E}_O[\bar{f}_\theta(x)|S])\right] \\ &= \mathbb{E}_x\left[\text{Var}_S(\mathbb{E}_O[f_\theta(x)|S])\right]\end{aligned} \tag{8}$$

which indicates that $\mathcal{E}_{var_O}$ reduces while $\mathcal{E}_{var_S}$ keeps unchanged as the ensemble size N increases. $\mathcal{E}_{var_O}$ vanishes when N is sufficiently large. The improvement of the variance by ensembling comes from the reduction of the variance due to optimization.

As mentioned in Section 2.2 that under linear squared regression assumption, $\mathcal{E}_{var_O}$ vanishes as p increases and $\mathcal{E}_{var_S}$ depends on critical parameter dimensionality $d(p)$. In this paper, we also proved that $\mathcal{E}_{var_O}$ vanishes as N increases. Given that pretraining LPMs with larger p and/or q is extremely difficult, increasing N is a much better way for improving performance of LPMs. This also proves the effectiveness of Stage-II in our proposed two-stage fine-tuning method. To ensure that a fine-tuned LPM can move from Phase-I to Phase-II, the ensemble size N in Stage-II should be set to a value that is larger or equal to $\frac{\mathcal{E}_{var_O}}{\mathcal{E}_{var_S}}$.



(a) F1 absolute improvement

(b) F1 percentage improvement

Figure 2: Performance improvement of low-resource (NOTED-ON, BEGINS-ON, ENDS-ON) and high-resource classes (CONTAINS, OVERLAP, BEFORE) on THYME data. We show absolute improvement (a) and improvement percentage (b) (compared with single learners i.e., $N = 1$) with F1.

5.3 Ensemble in low- and high-resource classes

One interesting experimental observation is that the improvement on low-resource classes is larger than that on high-resource classes. To further investigate the impact of the ensemble learners on the imbalanced datasets, we make the following analysis to Equation 5.

$$\mathbb{E}_x \text{Var}(\bar{f}_\theta(x)) = \begin{cases} \frac{1}{N} \cdot \frac{p}{m}\sigma_\epsilon^2 & \text{for} \quad p \leq m \\ \frac{1}{N} \cdot \frac{r}{m}\sigma_\epsilon^2 & \text{for} \quad p > m \end{cases} \tag{9}$$

where the impact of ensembles is represented as a function of ensemble size N, denoted as $\pi(N) = \frac{1}{N} \in [0, 1]$ that a smaller $\pi(N)$ means a larger performance improvement. Given a fixed p, in over-parameterized cases ($p > m$) where m is small, since the samples in S are i.i.d, thus $X \in \mathbb{R}^{m \times p}$ is full rank so that $r = \text{rank}(X) = m$. The variance becomes $\frac{1}{N} \cdot \sigma_\epsilon^2$ which does not change with m. The impact of ensemble solely depends on N thus is significant. On the other hand, in under-parameterized cases ($p \leq m$) where m is large, the variance is negligible as m becomes much larger than p, i.e., $\lim_{m \to \infty} \frac{1}{N} \cdot \frac{p}{m}\sigma_\epsilon^2 = 0$, so that the variance becomes 0 regardless of $\pi(N)$. This implies that the impact of ensemble can be ignored as m increases. In general, given a fixed p, for both cases the impact of ensemble is significant when m is small and insignificant as m becomes very large.

These theoretical findings explain why we observe larger performance improvement on low-resource than on high-resource classes using ensembles. Similar to the discussion in Section 4.4, empirically, it is hard to define low-resource and high-resource classes using m and p because our analysis is based on least squared linear regression assumption which is simplified compared to conditions in real scenarios. Besides p and m, there are other factors that may have implicit but significant impact on model performance. This also explains why the improvement does not strictly follow the sorting of classes by their sample size. However, our findings show another advantage of using ensembles. The empirical impact of ensemble size on imbalanced classes has been examined and shown in Section 4.6 and Appendix C, which is consistent with the theoretical findings discussed in this section.

6 Related Works

In a fine-tuning setting, searching for an optimal setting of pretrained models and hyperparameters is challenging due to the high dimensionality of the search space, as well as the infinite values for each dimension. In previous works of fine-tuning tasks (Lee et al., 2020; Alsentzer et al., 2019; Beltagy et al., 2019; Lin et al., 2021), the SOTA models are single learners carefully selected and fine-tuned based on evaluation results, such as P, R, and F1 scores, using grid-search or random-search. To improve the stability of the pre-trained transformer-based language models, Mosbach et al. (2021) suggests using small learning rates with bias correction and increasing the number of iterations to avoid vanishing gradients. Prior efforts also highlight the comparable effect of weight initialization and training data order to the variance of model performance (Dodge et al., 2020).

Ensemble methods have been successful in boosting the predictive performance of single learners (Ren et al., 2016 present a comprehensive review; also see Wang et al., 2003; CireşAn et al., 2012; Xie et al., 2013; Huang et al., 2017) as well

as in estimating predictive uncertainty (Gal and Ghahramani, 2016; Lakshminarayanan et al., 2017; Snoek et al., 2019). Among these studies, Bonab and Can (2019) and Wang et al. (2020) theoretically prove that ensembles can perform better than the average performance of component learners for prediction accuracy and consistency of learning models. Wang et al. (2022) empirically evaluates the application of ensemble methods to fine-tuned transformer-based models for clinical NLP tasks. The findings demonstrate that ensemble methods improve model performance, particularly when employing dynamic snapshot ensembling. Although it is common knowledge that ensembles can reduce variance thus reducing the generalization error, no previous work has discussed or measured this in the context of variance decomposition. Furthermore, no previous work has investigated the impact of ensembles on imbalanced datasets.

7 Conclusion

Different from the prevailing fine-tuning settings, we propose a two-stage fine-tuning method to improve the generalization ability of a fine-tuned LPM. We provide a variance decomposition-based justification method to empirically measure the generalization ability of the LPM w.r.t. the upper limit of its learning ability. In Stage-I, the RSOTA setting is built by ratcheting down bias and variance in an iterative way. In Stage-II, given the RSOTA setting, the fine-tuned LPM is guaranteed to be further generalized through ensembling techniques by reducing the variance due to optimization. The proposed justification method provides a concrete metric to track this process.

We provide empirical evidence by conducting experiments on the SuperGLUE tasks and two clinical datasets. Furthermore, we perform theoretical analysis on how ensembles improve variance due to optimization. We investigate the nature of variance change for the ensemble size in low- and high-resource classes in classification tasks. Different from previous theoretical analyses using only model complexity and data size which depends on least squared regression, our variance decomposition-based justification method in Stage-I does not rely on specific factors thus leading to a more generalizable measurement. The ENS further boosts performance without risk of computational cost and overfitting. Our analysis on imbalanced data reveals another advantage of ensemble algorithms in improving model performance on low-resource classes.

As future work, we are interested in (1) rigorously proving variance decomposition-based justification criteria, (2) quantifying low- and high-resource classes with specific features that interplay with ensemble size. If properly used, we believe the theoretical and empirical findings discussed in this paper can guide practitioners to fine-tune more generalizable models.

Limitations

As we stated under future work, one of the limitations is the variance decomposition-based proof. Our work is based on simplified settings, i.e., linear squared regression assumption. Post-ensemble variance is not evaluated due to the nature of the ENS ensemble algorithm. Extended experiments using vanilla bagging ensemble would enable analysis of post-ensemble variance. Further investigation into refining the two stages would help understand the performance of LPMs, e.g. those that are in Phase-I but before the peak in Figure 1. Our results for MultiRC are based on the instance sampling, however a better sampling technique should be based on stratified sampling based on the ratio of the question types in the MultiRC set. However, to achieve this, the MultiRC set needs to be annotated for question types, which is currently missing. Sampling techniques by themselves can become a research topic so that a further decrease of variance due to sampling can be achieved. Although we list these items as limitations, they are also topics for future research within the greater theme of understanding the new bias-prevalence paradigm for LPMs.

Acknowledgements

The authors would like to thank the anonymous reviewers for feedback that improved the paper, the US National Institutes of Health (NIH) and the New Jersey Institute of Technology (NJIT) for providing funding. This research is supported by NJIT FY24 Faculty Seed Grant, NIH Grants U24CA248010, R01LM013486 and R01GM114355. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NJIT or NIH.

References

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

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

Hamed Bonab and Fazli Can. 2019. Less is more: a comprehensive framework for the number of components of ensemble classifiers. *IEEE Transactions on neural networks and learning systems*.

Dan CireşAn, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. 2012. Multi-column deep neural network for traffic sign classification. *Neural networks*, 32:333–338.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059.

Stuart Geman, Elie Bienenstock, and René Doursat. 1992. Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58.

Hasham Ul Haq, Veysel Kocaman, and David Talby. 2021. Deeper clinical document understanding using relation extraction. *arXiv preprint arXiv:2112.13259*.

Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E Hopcroft, and Kilian Q Weinberger. 2017. Snapshot ensembles: Train 1, get m for free. *arXiv preprint arXiv:1704.00109*.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. 2017. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pages 6402–6413.

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

Chen Lin, Timothy Miller, Dmitriy Dligach, Steven Bethard, and Guergana Savova. 2021. EntityBERT: Entity-centric masking strategy for model pretraining for the clinical domain. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 191–201, Online. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2021. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. In *9th International Conference on Learning Representations*, CONF.

Brady Neal, Sarthak Mittal, Aristide Baratin, Vinayak Tantia, Matthew Scicluna, Simon Lacoste-Julien, and Ioannis Mitliagkas. 2018. A modern take on the bias-variance tradeoff in neural networks. *arXiv preprint arXiv:1810.08591*.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning.

Ye Ren, Le Zhang, and Ponnuthurai N Suganthan. 2016. Ensemble classification and regression-recent developments, applications and future directions. *IEEE Computational Intelligence Magazine*, 11(1):41–53.

Jasper Snoek, Yaniv Ovadia, Emily Fertig, Balaji Lakshminarayanan, Sebastian Nowozin, D Sculley, Joshua Dillon, Jie Ren, and Zachary Nado. 2019. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13969–13980.

William F. Styler IV, Steven Bethard, Sean Finan, Martha Palmer, Sameer Pradhan, Piet C de Groen, Brad Erickson, Timothy Miller, Chen Lin, Guergana Savova, and James Pustejovsky. 2014. Temporal annotation in the clinical domain. *Transactions of the*

Association for Computational Linguistics, 2:143–154.

Weiyi Sun, Anna Rumshisky, and Ozlem Uzuner. 2013. Evaluating temporal relations in clinical text: 2012 i2b2 challenge. *Journal of the American Medical Informatics Association*, 20(5):806–813.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. 2003. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 226–235. AcM.

Lijing Wang, Dipanjan Ghosh, Maria Gonzalez Diaz, Ahmed Farahat, Mahbubul Alam, Chetan Gupta, Jiangzhuo Chen, and Madhav Marathe. 2020. Wisdom of the ensemble: Improving consistency of deep learning models. *Advances in Neural Information Processing Systems*, 33:19750–19761.

Lijing Wang, Timothy Miller, Steven Bethard, and Guergana Savova. 2022. Ensemble-based fine-tuning strategy for temporal relation extraction from the clinical narrative. In *Proceedings of the 4th Clinical Natural Language Processing Workshop*, pages 103–108, Seattle, WA. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Kristin Wright-Bettner, Chen Lin, Timothy Miller, Steven Bethard, Dmitriy Dligach, Martha Palmer, James H. Martin, and Guergana Savova. 2020. Defining and learning refined temporal relations in the clinical narrative. In *Proceedings of the 11th International Workshop on Health Text Mining and Information Analysis*, pages 104–114, Online. Association for Computational Linguistics.

Notation	Description
$\mathcal{X}, \mathcal{Y}$	The input and output sets of a learning task.
$\mathcal{D}$	The unknown joint distribution of $(\mathcal{X}, \mathcal{Y})$.
(x, y)	The pair drawn from $\mathcal{D}$; $x \in \mathcal{X}, y \in \mathcal{Y}$.
S	A finite training dataset of m i.i.d. samples from $\mathcal{D}$.
m	The sample size of S.
C	The number of classes in a training dataset.
p	The number of hidden units in a neural network layer.
q	The number of hidden layers in a neural network.
f_θ	The predictors that are parameterized by the weights $\theta \in \mathbb{R}^p$ of neural networks.
$f_\theta(x)$	The output prediction given x.
O	The random variable for optimization randomness.
$\mathcal{A}$	The learning algorithm that produces $\theta = \mathcal{A}(S, O)$.
$\mathbb{E}[y\|x]$	The expectation of y given x.
$\mathcal{R}_m$	The performance of a learning algorithm using training sets of size m.
$\mathcal{E}_{noise}$	The expected noise of the output predictions.
$\mathcal{E}_{bias}$	The expected bias of the output predictions.
$\mathcal{E}_{var}$	The expected variance of the output predictions.
Var_{opt}	The variance due to optimization.
Var_{samp}	The variance due to sampling.
$\mathcal{E}_{var_O}$	The expected variance due to optimization.
$\mathcal{E}_{var_S}$	The expected variance due to sampling.
N	The ensemble size.
$\mathbb{G}_p$	Generalization ability of a learning algorithm.

Table 3: Notations and their descriptions.

Jingjing Xie, Bing Xu, and Zhang Chuang. 2013. Horizontal and vertical ensemble with deep representation for classification. *arXiv preprint arXiv:1306.2759*.

Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, and Yi Ma. 2020. Rethinking bias-variance trade-off for generalization of neural networks. In *International Conference on Machine Learning*, pages 10767–10777. PMLR.

Yian Zhang, Alex Warstadt, Haau-Sing Li, and Samuel R Bowman. 2020. When do you need billions of words of pretraining data? *arXiv preprint arXiv:2011.04946*.

A Notations

Table 3 shows major notations and their descriptions.

B Experiments

B.1 Data description

Table 4 shows the statistics of all datasets used in our experiments. Each downloaded SuperGLUE dataset includes train, val, and test sets in json format.[2] The downloaded test set does not have gold-standard labels thus is not used in our experiment.

[2] https://super.gluebenchmark.com/tasks

	BoolQ	CB	RTE	MultiRC	WiC	COPA	THYME	i2b2
Classes	2	3	2	2	2	2	10	3
Train samples	7541	200	2000	4080	4800	320	423612	9909
Dev samples	1886	50	500	1020	1200	80	235117	2478
Val samples	3270	57	278	953	638	100	208892	9630

Table 4: Data statistics.

	BoolQ	CB	RTE	MultiRC	WiC	COPA	THYME	i2b2
Random seed	62	52	72	72	42	72	42	42
Batch size	10	10	10	10	10	10	32	32
Epoch	8	7	10	6	8	8	3	3
Learning rate	1e-5	2e-5	2e-5	2e-5	1e-5	1e-5	4e-5	4e-5
Learning rate schedule type	*linear*	*linear*	*linear*	*linear*	*linear*	*linear*	*linear*	*linear*
Max sequence length	512	512	512	512	512	512	100	128
Gradient accumulation steps	2	2	2	2	2	2	2	2

Table 5: The RSOTA settings for SuperGLUE tasks and clinical information extraction tasks.

We split the train set into train (80%) and dev (20%) sets, and evaluate the model performance on val set. The i2b2 does not have a development (dev) set in the released data and we split the train set into train (80%) and dev (20%) sets. Random seed 42 is used to replicate the sampling process. For MultiRC, because each question can have more than one correct answer, we sampled the instances based on individual question-answer options in the train set for training and validation in our experiment.

B.2 Hyperparameter settings

Table 5 shows the details of hyperparameter settings. Unless otherwise specified, we use default values of the hyperparameters in Huggingface. We also summarize pretrained models used in our experiments in Table 6.

B.3 Replicated SOTA scores

To ensure that our experiments on the SuperGLUE tasks are reproducible, we followed the settings and replicated the SOTA accuracy scores reported in: https://github.com/facebookresearch/fairseq/tree/main/examples/roberta. We could not replicate the representation (special token extractions) and the model settings (unpublished pretrained model) for WSC task, thus it is omitted in our paper. In our experiments, we report the classic metrics of precision/recall/F1 for consistency with the reported results for the THYME and i2b2 tasks. Our accuracy scores for the SuperGlue tasks (shown in Table 7) are directly comparable and are consistent with those in Table 2 in the main paper.

B.4 Implementation details of ENS

ENS allows a pretrained model to be fine-tuned multiple times (i.e., multiple training runs) sequentially with different random seeds and data shuffling of train/validation splits. It uses a cyclic annealing schedule and cyclic snapshot strategy to periodically save the best model during each training run. Different from the simple bagging ensemble, after each training run, a dynamic pruning algorithm is applied to select a few single learners from the saved ones which can lead to better performance of the ensemble learner with theoretical guarantees. The sequential training runs stop when the accumulated number of selected single learners reaches a preset ensemble size. The total amount of training runs is a dynamic value rather than a preset value, which is determined by the snapshot strategy and pruning factor during the sequential training.

In our experiments, we implemented ENS on the top of RSOTA setting. The ensemble size is set as 5 and majority voting is used to generate ensemble predictions. We reuse RSOTA settings except that we set *cosine with restarts* as the learning rate scheduler and set the learning rate to restart every k epochs which, based on the RSOTA setting, allows the model to converge to a reasonable state before each restart. The total number of epochs for each training run is $5 \times k$ and we save the top 4 models for pruning based on validation accuracy. The random seeds for initialization and data shuffling are [42, 52, 62, 72, 82]. The logic behind the above settings is to retain the benefits from RSOTA fine-tuning settings as much as possible. Code and settings to reproduce the results are avail-

Model name	Model Details
RoBERTa-base	12-layer, 768-hidden, 12-heads, 125M parameters.
RoBERTa-large	24-layer, 1024-hidden, 16-heads, 355M parameters
PubmedBERTbase-MimicBig-EntityBERT	12-layer, 768-hidden, 12-heads, 110M parameters.
BioBERT-base-cased-v1.1	12-layer, 768-hidden, 12-heads, 110M parameters.

Table 6: Details of pretrained models.

	BoolQ	CB	RTE	MultiRC	WiC	COPA
Reference	86.9	98.2	89.5	85.7	75.6	94.0
Replicated	86.3	98.2	87.4	84.7	72.1	93.4
RSOTA	85.4±0.31	81.1±9.9	83.7±0.91	79.1±10.72	70.7±1.7	81.6±11.9
ENS	85.7±0.41	92.2±1.4	85.6±1.05	82.5±1.3	71.3±0.36	93.6±0.48

Table 7: Accuracy scores on the SuperGLUE tasks. For "Reference" and "Replicated": training on the original train set, validating and testing on the original dev set. For "RSOTA" and "ENS": training on 80% of the original train set, validating on 20% of the original train set, and testing on the original dev set.

able at https://github.com/christa60/bias-var-fine-tuning.git.

B.5 Experimental design of bagging ensemble for investigating various ensemble sizes

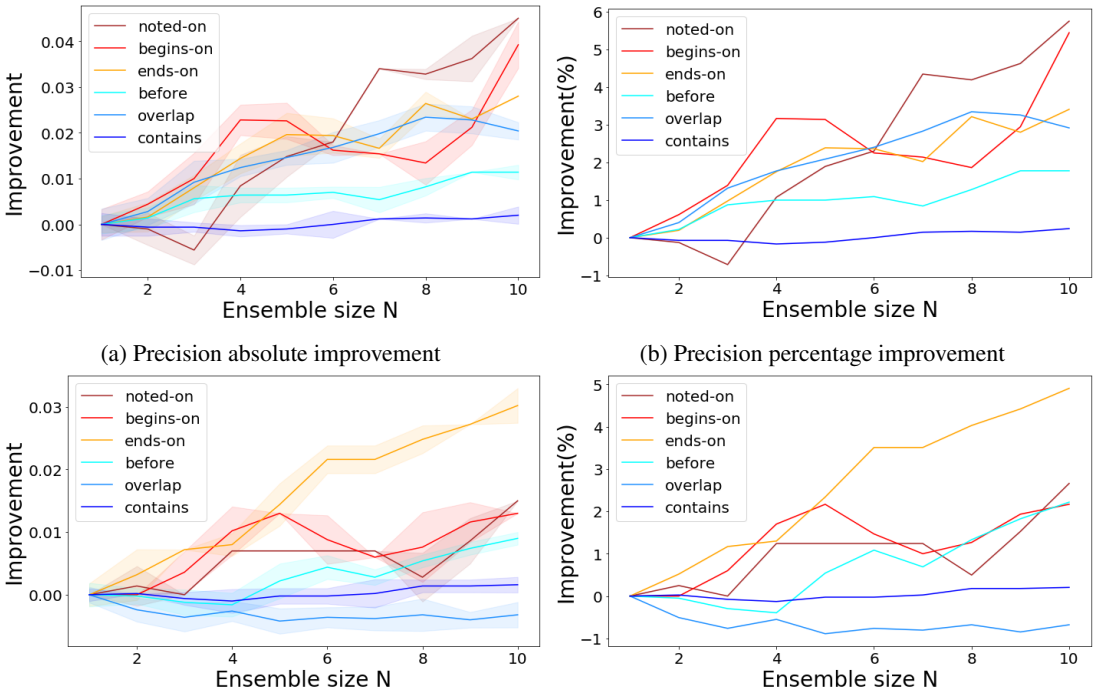
To analyze the nature of the variance change with the ensemble size in low-resource classes (NOTED-ON, BEGINS-ON, END-ON relations in the THYME corpus) and high-resource (CONTAINS, OVERLAP, BEFORE relations in the THYME corpus) classes, we vary the ensemble size from 1 to 10 and then compute the P, R, and F1 scores for each class on THYME data.

We create 10 bootstrap replicate training sets by resampling training and dev datasets with the same size and class distribution. The random seeds for resampling are randomly chosen and then fixed. The various splits are denoted as ['split_r42', 'split_r52', 'split_r62', 'split_r72', 'split_r82', 'split_r92', 'split_r102', 'split_r112', 'split_r122', 'split_r132']. Given a random seed of initialization, we train N fine-tuned single learners. To compute 95% confidence intervals for these estimates, we use 5 random seeds of initialization, resulting in 5 ensemble models for each ensemble size. We vary the ensemble size N from 1 to 10 and have 100 ensemble models in total.

C Section 4.6: Additional Results

We show the absolute and percentage improvement (compared with single learners i.e., $N = 1$) change over the ensemble size N using P and R in Figure 3. Together with Figure 2, the *major observations* are: (a) The absolute and percentage improvements of P, R, and F1 increase as N increases. (b) The precision improvements are more pronounced than those of recall thus contributing the major part of the F1 improvements. This phenomenon is more pronounced for high-resource classes. (c) Given a fixed N, the improvements on low-resource classes are larger than those on high-resource classes across the three metrics. The difference becomes larger as N increases.

Discussion: Our experimental results are consistent with our theoretical findings in Section 5 that model performance keeps improving because variance due to optimization decreases as ensemble size increases. Furthermore, the impact of ensemble is more pronounced on low-resource classes than on high-resource classes.



(a) Precision absolute improvement

(b) Precision percentage improvement

(c) Recall absolute improvement

(d) Recall percentage improvement

Figure 3: Performance improvement on low-resource classes (NOTED-ON, BEGINS-ON, ENDS-ON) and high-resource classes (CONTAINS, OVERLAP, BEFORE) from the THYME dataset. We show absolute and percentage improvement (compared with single learners i.e., $N = 1$) for precision ((a) and (b)), and recall ((c) and (d)). Values are computed based on the mean with 95% confidence interval over 5 random samples for each class and each ensemble size.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Limitations

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 1 Introduction

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4

☑ B1. Did you cite the creators of artifacts you used?
Section 4

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 4

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 4

C ☑ Did you run computational experiments?

Section 4

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 4 and Appendix B

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4 and Appendix B

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 4.4 and 4.5

☐ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Not applicable. Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.