FTFT: Efficient and Robust Fine-Tuning by Transferring Training Dynamics

Yupei Du, Albert Gatt, and Dong Nguyen
Utrecht University
Utrecht, the Netherlands
{y.du, a.gatt, d.p.nguyen}@uu.nl

Abstract

Despite the massive success of fine-tuning Pre-trained Language Models (PLMs), they remain susceptible to out-of-distribution input. Dataset cartography is a simple yet effective dual-model approach that improves the robustness of fine-tuned PLMs. It involves fine-tuning a model on the original training set (i.e. reference model), selecting a subset of important training instances based on the training dynamics, and fine-tuning again only on these selected examples (i.e. main model). However, this approach requires fine-tuning the same model twice, which is computationally expensive for large PLMs. In this paper, we show that 1) training dynamics are highly transferable across model sizes and pre-training methods, and that 2) fine-tuning main models using these selected training instances achieves higher training efficiency than empirical risk minimization (ERM). Building on these observations, we propose a novel fine-tuning approach: Fine-Tuning by transFerring Training dynamics (FTFT). Compared with dataset cartography, FTFT uses more efficient reference models and aggressive early stopping. FTFT achieves robustness improvements over ERM while lowering the training cost by up to $\sim 50\%$.[1]

1 Introduction

Despite the success of few-shot and zero-shot learning (Brown et al., 2020), state-of-the-art performance in Natural Language Processing (NLP) is still largely achieved by fine-tuning large Pre-trained Language Models (PLMs) (Mosbach et al., 2023). Scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022) suggest that better downstream performance is achieved with larger PLMs. However, fine-tuning large PLMs is also more expensive, in terms of both computational resources and carbon emission (Strubell et al., 2019; Wu et al., 2022).

Moreover, despite impressive progress on regular benchmarks, many studies have shown that fine-tuned PLMs lack robustness against out-of-distribution (OOD) input. For instance, human annotators can easily exploit the weaknesses of fine-tuned PLMs to trick these models to yield incorrect predictions, on tasks such as Natural Language Inference (NLI) (Nie et al., 2020) and Hate Speech Detection (HSD) (Vidgen et al., 2021b).

The problem of robustness can be mitigated using dual-model approaches. With such approaches, first a **reference model** is trained to estimate the importance of each training instance, and then a **main model** is trained based on the outputs of the reference model (Nam et al., 2020; Utama et al., 2020; Sanh et al., 2021; Karimi Mahabadi et al., 2020; Zhang et al., 2022; Liu et al., 2021). Among these approaches, dataset cartography (Swayamdipta et al., 2020) is especially attractive in view of its simplicity and the consistent improvements in model robustness. It consists of three steps. First, a **Data Map (DM)** is constructed from the **training dynamics** (i.e. instance prediction probabilities) of a fine-tuning run of the reference model on the full dataset. This DM divides the training data into three subsets: ambiguous, hard-to-learn, and easy instances. Finally, the main model is fine-tuned using only either the ambiguous or hard-to-learn subset. An important question has to do with the choice of reference model. Swayamdipta et al. (2020) use the same PLM for both the reference and main model. However, a major drawback of dataset cartography is the high computational cost, because it requires fine-tuning the same model twice. In this paper, *we jointly address robustness and efficiency issues* without sacrificing the simplicity of dataset cartography, by exploiting the **transferability of training dynamics**. We make three contributions.

First, we study the following question: Are DMs transferable across different model sizes and pre-training methods? We focus on the novel setting

---

[1]Our code is publicly available at https://github.com/nlpsoc/FTFT.

where *DMs are constructed based on computationally efficient reference models to fine-tune more capable — and often larger — main models*. Our motivation is two-fold: 1) efficient reference models ease the computational burden of constructing DMs, and 2) less capable reference models might be better at identifying ambiguous or hard training instances, because they are less likely to memorize training data (Tirumala et al., 2022; Carlini et al., 2023). Our results show that, in most cases, *training dynamics are highly transferable* across different model sizes (§4.1) and pretraining methods (§4.2). We further show that the condition for successful transfers is a reasonably strong reference model, which we also make precise in §4.3.

Second, we observe that fine-tuning with selected instances achieves consistently higher training efficiency than conventional fine-tuning (§6).

Third, building on these findings, we propose **Fine-Tuning by transFerring Training dynamics (FTFT, §6)**: an efficient fine-tuning approach that leads to improved OOD performance. Compared to dataset cartography, FTFT uses *efficient reference models* and *early stopping*. Experiments on two tasks, NLI and HSD, show that FTFT achieves better performance on OOD input than conventional Empirical Risk Minimization (ERM), while lowering the training cost by up to $\sim 50\%$.

2 Background

Dual-Model Approaches for Robustness Many studies have proposed dual-model approaches to improve model robustness that do not require knowledge of identifiable subsets of the data samples, such as NLI pairs in which hypotheses contain a negation (Gururangan et al., 2018), or HSD samples targeting a specific group (Dixon et al., 2018; Park et al., 2018). Nam et al. (2020) first train a reference model using generalized cross-entropy loss, and then train a main model while assigning higher weights to instances that were hard for the reference model. Sanh et al. (2021) use a Product-of-Expert (PoE) approach, by first training a reference model with limited capacity to capture dataset biases, and then training the main model to avoid these biases using PoE loss. Liu et al. (2021) propose the Just-Train-Twice approach (JTT), which involves first training a weak reference model using heavy regularization and vanilla SGD, and then upweighing the training instances that the reference model predicts incorrectly when training the main

model. Dataset cartography (Swayamdipta et al., 2020) is based on a similar idea, but it uses training dynamics instead of correctness to categorize training instances. We discuss this method below.

Dataset Cartography is a dual-model approach to improve model robustness. First, a reference model is trained on the full training dataset. Then, a Data Map (DM) is built based on the training dynamics, by tracking prediction probabilities of the true class (p_{true}) of each training instance across epochs. DM categorizes training instances into three subsets: *ambiguous* (i.e. the variance of p_{true} is in the top $q\%$ of all training instances); *hard-to-learn* (i.e. the mean of p_{true} is in the bottom $q\%$ of all training instances); and *easy* (i.e. neither ambiguous nor hard-to-learn). The threshold $q\%$ is fixed and typically set to 33%. Note that a training instance can be categorized as both hard-to-learn and ambiguous (a low mean but high variance for p_{true}). Finally, the main model is fine-tuned only on the ambiguous or hard-to-learn subset. Swayamdipta et al. (2020) show that, with a slight loss of In-Distribution (ID) performance, dataset cartography improves Out-Of-Distribution (OOD) performance of models. In this paper, we train main models with ambiguous data, since Swayamdipta et al. (2020) reported better performance using this data than hard-to-learn data.

Swayamdipta et al. (2020) use the same PLM as both the reference and the main model. In contrast, Sar-Shalom and Schwartz (2023) show that a DM constructed by ELECTRA$_{\text{Large}}$ (Clark et al., 2020) can be used to improve the robustness of DeBERTaV3$_{\text{Large}}$ (He et al., 2023). However, instead of using only the ambiguous subset, they added k copies of this subset to the original training set to train the main model. Moreover, they did not investigate either DM transfer across model sizes and pretraining methods, or how such transferability can be exploited to improve efficiency.

Model-Based Data Selection/Reweighing Our work is also connected to studies that have selected or reweighed data using reference models to improve in-distribution (ID) performance. Chang et al. (2017) use p_{true} variance and proximity to the classification threshold from a reference model to reweigh training instances; Toneva et al. (2019) calculate the frequency of forgetting events (i.e. from a correct to incorrect prediction), and remove the least forgettable instances; Paul et al. (2021) and Baldock et al. (2021) instead use error vector

norm and effective prediction depth to estimate the contribution of a training instance.

Previous studies have also explored the use of a smaller reference model to improve efficiency. Coleman et al. (2020) use a small model for active learning and core-set selection. Xie et al. (2023) reweigh domains for language model pretraining, by training a small reference model to estimate the difficulty of each domain.

3 Experimental Setup

We perform our experiments on two tasks, Natural Language Inference (NLI) and Hate Speech Detection (HSD). Following Swayamdipta et al. (2020), we select 33% the most ambiguous datapoints.

Data To study model robustness, we include challenging OOD test sets for each task, in addition to the training set and an ID validation set. For NLI, we use the MultiNLI dataset (Williams et al., 2018) as the train and ID validation set, because of its diverse composition, covering 10 genres. We use AdversarialNLI (Nie et al., 2020) as the OOD test set, which consists of three rounds of adversarial data collection. AdversarialNLI is known to be challenging and it targets weaknesses of models trained on MultiNLI.

For HSD, we use CAD (Vidgen et al., 2021a) as the training and ID validation set. CAD consists of Reddit posts covering diverse topics and writing styles, annotated using a fine-grained taxonomy. Following Ramponi and Tonelli (2022), we frame it as a binary classification task, marking identity-related abuse as hateful and other categories as non-hateful. As OOD test sets, we use DynaHate (Vidgen et al., 2021b), since it aligns with CAD's definition of hate speech. DynaHate contains three rounds of adversarial data collection and perturbations.

Models We mainly use DeBERTaV3 (He et al., 2023) and ELECTRA (Clark et al., 2020) in our experiments, due to their strong performance and availability in multiple model sizes. To study the transferability across different pretraining methods, we also use TinyBERT (Turc et al., 2020), BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) as reference models.[2] Besides ERM, we include a baseline that uses a random DM (i.e.,

random $q\%$ of the training data). We perform a search of optimal training steps on ERM, and use the same hyper-parameters for both ERM and data cartography approaches. Full details of the models and training setups are in Appendix A.1.

4 Transferability of Training Dynamics

In this section, we study the transferability of training dynamics in dataset cartography, i.e., whether we can use different reference and main models while maintaining the robustness advantage of the main model. Specifically, we study whether training dynamics are transferable across different model sizes (§4.1, e.g., from DeBERTaV3$_{Small}$ to DeBERTaV3$_{Large}$) and pretraining methods (§4.2, e.g., from ELECTRA$_{Large}$ to DeBERTaV3$_{Large}$).

We focus on these issues for two reasons. First, transferability across model sizes enables using more efficient (and usually less capable) reference models, which (1) can improve training efficiency, and (2) is potentially more effective in identifying ambiguous/hard instances, because they are usually worse at memorizing training instances. Second, transferability across pretraining methods can help achieve these advantages even in cases where more efficient variants of the main pretraining method are unsuitable or unavailable for the task. Moreover, understanding transferability can shed light on data importance: If DMs of different reference models consistently identify the same subset of training instances as ambiguous, it suggests that DMs reveal intrinsic data characteristics.

We define successful transfers as transfers that produce comparable or better OOD performance than ERM. Our results show that, with a few exceptions, training dynamics are indeed transferable. To understand the conditions for successful transfers, we analyze the failure cases (§4.3). We find that the DMs of reference models that lead to successful transfers typically identify a larger subset as easy, which serves as a rough indicator of their capability. This finding can serve as a guideline for choosing reference models without training the main model, which is computationally expensive.

4.1 Transferability Across Model Sizes

In this section, we study whether smaller and more efficient models can be used as reference models to construct DMs for training larger main models (e.g. DeBERTaV3$_{Small}$ as the reference model for DeBERTaV3$_{Large}$). Levering transferability of this

---

[2] Costs for fine-tuning different PLMs are in Appendix A.1. We report FLOPs rather than GPU hours because we noticed occasional low GPU utilization especially when fine-tuning smaller PLMs.

Method	Main Model	Ref. Model	Cost	MultiNLI	AdversarialNLI (Test)		
				-	R1	R2	R3
Baselines: ERM, ERM with Early Stopping, and Random DM							
ERM	DeBERTaV3$_{Small}$	-	14.47	$87.76_{0.09}$	$33.25_{1.67}$	$30.07_{0.71}$	$31.89_{0.46}$
ERM	DeBERTaV3$_{Base}$	-	28.29	$90.03_{0.14}$	$43.73_{0.66}$	$33.95_{0.53}$	$33.79_{1.20}$
ERM	DeBERTaV3$_{Large}$	-	100.00	$91.15_{0.08}$	$59.90_{1.95}$	$45.10_{1.39}$	$42.08_{0.95}$
ERM(ES)	DeBERTaV3$_{Large}$	-	66.67	$91.30_{0.29}$	$59.20_{1.28}$	$44.35_{1.66}$	$40.29_{0.98}$
DM	DeBERTaV3$_{Large}$	Random	100.00	$90.66_{0.23}$	$55.05_{0.78}$	$43.83_{0.72}$	$39.48_{0.43}$
Training Dynamics Transferability: Across Different Model Sizes							
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Large}$	200.00	$90.92_{0.16}$	$59.02_{1.97}$	$46.08_{2.37}$	$41.85_{0.30}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Small}$	114.47	$90.77_{0.17}$	$60.10_{1.58}$	$46.23_{0.44}$	$41.46_{1.01}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Base}$	128.29	$90.64_{0.24}$	$59.67_{1.19}$	$45.88_{1.09}$	$43.00_{1.56}$
Training Dynamics Transferability: Across Different Pretraining Methods							
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Small}$	104.61	$90.84_{0.03}$	$50.68_{1.67}$	$39.92_{0.59}$	$37.10_{0.94}$
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Base}$	136.18	$90.28_{0.21}$	$60.75_{1.06}$	$47.30_{0.74}$	$42.92_{0.94}$
DM	DeBERTaV3$_{Large}$	RoBERTa$_{Large}$	216.78	$90.26_{0.06}$	$61.02_{1.11}$	$46.85_{0.58}$	$42.65_{0.89}$
DM	DeBERTaV3$_{Large}$	BERT$_{Large}$	213.49	$89.37_{0.15}$	$62.00_{0.92}$	$48.60_{0.65}$	$44.73_{0.48}$
FTFT: Efficient Reference Models + Aggressive Early Stopping							
FTFT	DeBERTaV3$_{Large}$	DeBERTaV3$_{Small}$	51.97	$90.74_{0.12}$	$59.38_{1.56}$	$45.80_{2.55}$	$42.38_{2.31}$
FTFT	DeBERTaV3$_{Large}$	DeBERTaV3$_{Base}$	74.12	$90.54_{0.29}$	$59.80_{1.88}$	$45.85_{1.48}$	$42.39_{1.25}$
FTFT	DeBERTaV3$_{Large}$	ELECTRA$_{Base}$	79.93	$90.03_{0.57}$	$60.75_{1.33}$	$47.35_{2.28}$	$44.17_{0.82}$

Table 1: Our results with DeBERTaV3 as the main model on NLI (measured by accuracy scores), which consist of four parts: (1) Baselines: DeBERTaV3 of different sizes trained using ERM, DeBERTaV3$_{Large}$ trained using ERM with early stopping (ERM(ES)), and DeBERTaV3$_{Large}$ trained using random DM (random 33% of the training data); (2) Training dynamics transferability across different sizes: training DeBERTaV3$_{Large}$ as the main model, using DMs constructed by different sizes of DeBERTaV3 as reference models; (3) Training dynamics transferability across different pretraining methods: training DeBERTaV3$_{Large}$ as the main model, using DMs constructed by different pretraining methods as reference models (ELECTRA$_{Small/Base}$, BERT, and RoBERTa); (4) FTFT (§6): training DeBERTaV3$_{Large}$ using our approach FTFT (§6), with DMs constructed by different reference models, as well as aggressive early stopping. R1–R3 in AdversarialNLI refer to different rounds of data collection. Cost column contains the relative computational cost of each method, compared to training DeBERTaV3$_{Large}$ using ERM. For example, the compute of DeBERTaV3$_{Base}$ with ERM is 28.29, meaning its compute costs are 28.29% of training DeBERTaV3$_{Large}$ using ERM. We observe that: (1) Training dynamics are transferrable across different sizes and pretraining methods, as constructing DMs using different reference models results in comparable performance; (2) FTFT achieves robustness improvements over ERM, while lowering the training cost by up to $\sim 50\%$.

(a) Consistency across different sizes
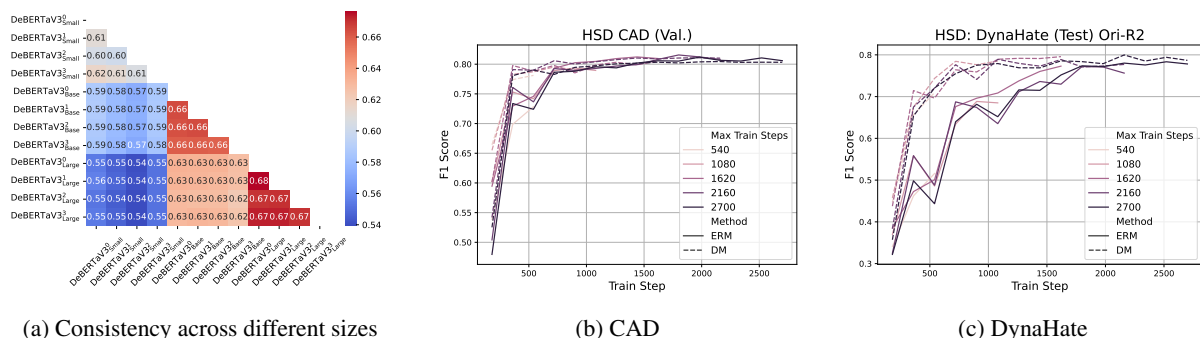

(b) CAD

(c) DynaHate

Figure 1: Figure 1a: Consistency across different sizes of DeBERTaV3 on NLI. The numbers are the percentages (0–1) of ambiguous training instances shared by two models. Training dynamics are transferable across different model sizes: the percentages between models of different sizes are only slightly smaller than those between models of different random seeds (shown as superscript). Figures 1b & 1c: Performance on HSD when training the main model (DeBERTaV3$_{Large}$) using different numbers of training steps. We experimented with different lengths of training (max training steps), and different methods (using ERM and DM). Training with data instances selected by DMs achieves consistently higher training speed than ERM: for datasets on which training with DM achieves either better (OOD datasets, right) or worse (ID datasets, left) performance, models trained with DM outperform ERM with fewer training steps (i.e. the early stage of training, the leftmost part of the x-axis).

type can improve the efficiency of dataset cartography by reducing the cost of constructing DMs.

The results for DeBERTaV3 are shown in Table 1 under section *across different model sizes* (NLI) and Table 5 (HSD, Appendix B). *Training dynamics are transferable across different model sizes*: when using DeBERTaV3$_{Large}$ as the main model, changing the reference model to either DeBERTaV3$_{Small}$ or DeBERTaV3$_{Base}$ yields comparable or even better performance. This observation is consistent with our hypothesis that efficient models are more sensitive to ambiguous and difficult instances. As a result, *they can serve as alternative efficient reference models*. We also observe that, consistent with Swayamdipta et al. (2020), ERM achieves better ID performance (in the baselines section of Table 1), while data cartography performs comparably or better on OOD data. Note that hyper-parameters are tuned for ERM performance (see Appendix A.1), *making the training setup more favorable overall to ERM*.

To investigate transferability of DMs further, we analyze whether reference models of different sizes identify similar groups of ambiguous instances. Figure 1a shows the percentage of ambiguous instances shared by reference models of different sizes and random seeds. The percentages shared between different model sizes are only slightly smaller than those between the same size but different random seeds, providing further evidence for the transferability of DMs.[3]

4.2 Transferability Across Pretraining Methods

We now study the transferability of training dynamics across different pretraining methods. Successful transfers of this type enable the use of efficient reference models, when there is no efficient version of the main model that suits the downstream task.

The results for DeBERTaV3$_{Large}$ as the main model with different reference models are shown in Table 1 under section *across different pretraining methods* (NLI) and Table 5 (HSD, Appendix B). *Training dynamics are generally transferable across different pretraining methods*: DeBERTaV3$_{Large}$ achieves comparable performance using DMs constructed by different reference models in most cases. However, there is one exception: when using ELECTRA$_{Small}$ as the reference model, the performance is clearly worse

on the NLI OOD datasets than when using ERM. We hypothesize that ELECTRA$_{Small}$ is not strong enough for constructing effective DMs. We analyze this hypothesis further in §4.3 below.

4.3 When Do Transfers Fail?

We have shown that training dynamics are usually transferable across different model sizes and pretraining methods. We now study the conditions for successful transfers, by zooming in on two questions: 1) Can we use efficient but weak models as reference models? and 2) What are the differences between effective and ineffective reference models? Answers to these questions can guide the selection of efficient yet effective reference models.

Can we use efficient but weak models as reference models? To answer this question, we compare the performance of a wide range of methods, see Table 2 (NLI) and Table 6 (HSD, in Appendix B). We include models from three categories: First, a wide range of seven models trained using ERM (section *ERM Performance*): TinyBERT, the small and base versions of DeBERTaV3 and ELECTRA, RoBERTa$_{Large}$, and BERT$_{Large}$. Second, we use these ERM models as reference models to construct DMs, and use these DMs to fine-tune DeBERTaV3$_{Large}$ (section *When Do Transfers Fail*, first half). By using reference models with different sizes and pretraining methods to fine-tune the same main model, we can inspect the impact of reference model capability on transferability. Third, we also include the results of using ELECTRA$_{Large}$ as the main model (section *When Do Transfers Fail*, second half). By comparing results from different main models, we can better understand whether successful transfers originate from the compatibility between reference and main models, or the capability of the reference model itself. We make three observations.

First, weak reference models with poor ID performance, i.e., TinyBERT and ELECTRA$_{Small}$, lead to unsuccessful transfers. However, the reference models do not need to be as capable as the main model: slightly weaker reference models could be more useful, indicated by the strong OOD performance when using BERT$_{Large}$ as the reference model. Moreover, in these unsuccessful transfers, the main model OOD performance correlates with the reference model ID performance. For example, on NLI, TinyBERT has a lower ID performance than ELECTRA$_{Small}$, and both main mod-

---

[3]The expected overlap between two random DMs is 0.33.

Method	Main Model	Ref. Model	Compute	MultiNLI (Val.)	AdversarialNLI (Test)		
				-	R1	R2	R3
ERM Performance: Capabilities of Various Models							
ERM	TinyBERT	-	1.45	$67.29_{0.26}$	$23.50_{0.82}$	$28.30_{0.36}$	$30.69_{0.42}$
ERM	ELECTRA$_{Small}$	-	4.61	$82.08_{0.09}$	$23.82_{0.75}$	$28.93_{1.16}$	$30.54_{0.70}$
ERM	ELECTRA$_{Base}$	-	36.18	$88.47_{0.20}$	$35.45_{0.24}$	$30.95_{0.42}$	$31.27_{0.68}$
ERM	DeBERTaV3$_{Small}$	-	14.47	$87.76_{0.09}$	$33.25_{1.55}$	$30.07_{0.66}$	$31.89_{0.43}$
ERM	DeBERTaV3$_{Base}$	-	28.29	$90.03_{0.13}$	$43.73_{0.61}$	$33.95_{0.49}$	$33.79_{1.11}$
ERM	BERT$_{Large}$	-	113.49	$86.25_{0.26}$	$20.12_{0.85}$	$29.05_{1.02}$	$29.89_{0.57}$
ERM	RoBERTa$_{Large}$	-	116.78	$89.86_{0.09}$	$43.85_{0.64}$	$28.55_{1.07}$	$26.00_{1.04}$
When Do Transfers Fail: Using Reference Models with Different Capabilities							
DM	DeBERTaV3$_{Large}$	TinyBERT	101.45	$89.26_{0.16}$	$42.47_{0.87}$	$34.38_{0.66}$	$33.56_{0.52}$
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Small}$	104.61	$90.84_{0.03}$	$50.68_{1.55}$	$39.92_{0.55}$	$37.10_{0.87}$
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Base}$	136.18	$90.28_{0.19}$	$60.75_{0.98}$	$47.30_{0.68}$	$42.92_{0.87}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Small}$	114.47	$90.77_{0.16}$	$60.10_{1.46}$	$46.23_{0.41}$	$41.46_{0.94}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Base}$	128.29	$90.64_{0.22}$	$59.67_{1.10}$	$45.88_{1.01}$	$43.00_{1.44}$
DM	DeBERTaV3$_{Large}$	BERT$_{Large}$	213.49	$89.37_{0.15}$	$62.00_{0.92}$	$48.60_{0.65}$	$44.73_{0.48}$
DM	DeBERTaV3$_{Large}$	RoBERTa$_{Large}$	216.78	$90.26_{0.06}$	$61.02_{1.03}$	$46.85_{0.54}$	$42.65_{0.83}$
DM	ELECTRA$_{Large}$	TinyBERT	111.64	$88.30_{0.22}$	$35.37_{0.31}$	$32.60_{1.25}$	$30.78_{0.64}$
DM	ELECTRA$_{Large}$	ELECTRA$_{Small}$	114.80	$90.35_{0.19}$	$45.38_{0.67}$	$34.95_{0.90}$	$32.58_{1.16}$
DM	ELECTRA$_{Large}$	ELECTRA$_{Base}$	146.38	$89.96_{0.20}$	$55.40_{0.28}$	$42.00_{0.42}$	$36.71_{0.65}$
DM	ELECTRA$_{Large}$	DeBERTaV3$_{Small}$	124.67	$90.27_{0.13}$	$54.20_{0.57}$	$40.00_{1.13}$	$36.88_{0.06}$
DM	ELECTRA$_{Large}$	DeBERTaV3$_{Base}$	138.49	$89.94_{0.14}$	$53.63_{0.80}$	$41.33_{0.51}$	$36.33_{0.25}$
DM	ELECTRA$_{Large}$	BERT$_{Large}$	223.68	$88.83_{0.12}$	$57.10_{0.79}$	$43.27_{0.84}$	$38.72_{0.31}$
DM	ELECTRA$_{Large}$	RoBERTa$_{Large}$	226.97	$89.62_{0.13}$	$54.20_{0.57}$	$42.55_{0.78}$	$37.29_{0.65}$

Table 2: We use reference models with different capabilities (measured by their ERM accuracy) to construct DMs, and use them to train main models. The gray-shaded rows are (1) the main models in unsuccessful transfers and (2) their corresponding reference models. Successful transfer requires the reference model to be reasonably strong: reference models with clearly worse ID performance lead to degraded OOD performance for the main models.

els DeBERTaV3$_{Large}$ and ELECTRA$_{Large}$ have a lower OOD performance when using TinyBERT as the reference model compared to when using ELECTRA$_{Small}$ as the reference model.

Second, whether a transfer is successful or not depends mostly on the reference model, rather than the compatibility between the reference and the main models: The transfers from TinyBERT and ELECTRA$_{Small}$ to both main models are unsuccessful, and the transfers from other reference models to both main models are successful.[4]

Third, interestingly, ID performance in all transfers remains high. In other words, ineffective reference models only impact the OOD performance of the main models. For instance, the transfer from ELECTRA$_{Small}$ to DeBERTaV3$_{Large}$ yields the best accuracy on MultiNLI despite its OOD performance being relatively poor. We suspect that weak models with poor ID performance often identify easy training instances as ambiguous data.

While easy instances can be sufficient for obtaining satisfactory ID performance, they are insufficient for good OOD performance (Swayamdipta et al., 2020). We discuss this in detail below.
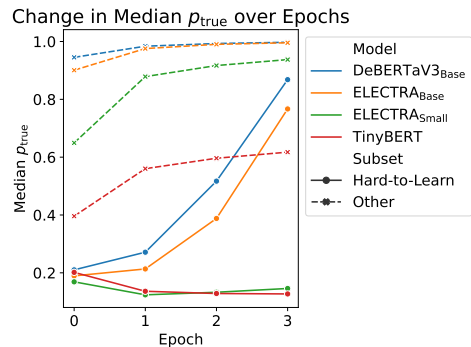

Figure 2: Change of median p_{true}: ineffective reference models (ELECTRA$_{Small}$ and TinyBERT) are unable to fit difficult training instances, making easy instances being identified as ambiguous.

What are the differences between effective and ineffective reference models? To answer this question, we consider the possible differences between a weak (and ineffective) reference model and a reasonably strong reference model, in terms

---

[4] We also include results using DeBERTaV3$_{Base}$ as the main model, and TinyBERT and DeBERTaV3$_{Small}$ as reference models on HSD in Table 6. Transfers from DeBERTaV3$_{Small}$ substantially outperform ERM, while those from TinyBERT underperform ERM, further suggesting that the reference model is decisive for successful transfers.

of categorizing training data into ambiguous, hard-to-learn, and easy subsets. Also, we assume that instances in our training set exhibit varying levels of difficulty (i.e. simple to difficult).

Assume we have a weak reference model that can learn simpler training instances but cannot learn the more difficult ones. This weak reference model will therefore assign increasing p_{true} to simple training instances across different epochs, while keeping p_{true} for difficult training instances around the values expected in a random guessing scenario. Consequently, p_{true} will exhibit high standard deviations on simpler training instances, which will then be identified as ambiguous data; while more difficult training instances will have consistent lower mean values and thus low standard deviations for p_{true}, and therefore be identified as hard-to-learn data. In contrast, a reasonably capable reference model can learn simpler training instances during the early stage of training (even before their first occurrence in the training batch, i.e., their correct predictions are learned from other training instances). Therefore, these instances will have both high mean values and low standard deviations for p_{true}. Meanwhile, p_{true} for difficult instances will gradually increase across epochs, making these instances yield relatively low mean values and high standard deviations for p_{true}. As a result, these instances will be identified as both ambiguous and hard-to-learn (i.e. we expect a large overlap in these subsets). Because we select a fixed percentage $q\%$ of instances as ambiguous or hard-to-learn, this larger overlap means a larger easy subset too.

We now validate our reasoning. Given a reference model, we first split the training instances into two subsets based on mean p_{true}: hard-to-learn (10% of training instances) and other (the remaining 90%). We use a lower $q\%$ to make the difference clearer. Then, for each subset, we calculate the median p_{true} in each epoch. We use median values because they are robust statistics of the central tendency. Figure 2 shows our results on MultiNLI, using two effective (DeBERTaV3$_{\text{Base}}$ and ELECTRA$_{\text{Base}}$) and two ineffective (ELECTRA$_{\text{Small}}$ and TinyBERT) reference models. We only include four models to make the figure clearer — we observe similar trends for other models: With effective reference models, hard-to-learn instances are gradually learned during training, while other instances already have high p_{true} values from the first epoch. In contrast, with ineffective reference models, hard-to-learn

data instances are not learned at all, suggested by their close-to-zero p_{true} over different epochs; while other instances are gradually fitted, indicated by their increasing values.

To further validate our reasoning, we compute the percentages of training instances identified as easy by different reference models (Table 4 in Appendix B): We report results with $q \in \{10\%, 25\%, 33\%, 50\%\}$. Less effective reference models indeed identify fewer data points as easy. For example, on NLI with $q\% = 50\%$, Tiny-BERT identifies less than 20.0% of the instances as easy, compared to 46.65% by DeBERTaV3$_{\text{Base}}$. Furthermore, the overlap between hard-to-learn and ambiguous instances in successful transfers is usually high. For example, with $q\% = 50\%$, all effective reference models identify more than 46% as easy training instances (the maximum is 50%, when the ambiguous and hard-to-learn subsets overlap perfectly).

5 Training Speed Gains From Data Selection

Our results from §4 suggest that the efficiency of dataset cartography can be improved with more efficient reference models. However, if we train the main models for the same number of steps as ERM, the computational cost of the full pipeline is still higher than conventional fine-tuning (i.e., ERM), because of the extra reference model training cost.

Previous studies have shown that training with a careful selection of "informative" training instances can lead to higher training speed than ERM (i.e. achieving the same performance with fewer training steps), despite the computational cost of selecting these instances being notoriously high (Sorscher et al., 2022; Feldman and Zhang, 2020; Paul et al., 2021; Toneva et al., 2019). Motivated by this, we now study whether we can achieve similar learning speed gains by training on instances selected by DMs. If such gains exist, we can further improve the efficiency of dataset cartography by training with fewer steps.

We show the performance of DeBERTaV3$_{\text{Large}}$ on HSD, fine-tuned using the full training set (ERM) and using only the instances selected by DMs, across different training durations (i.e. max training steps) in Figure 1. Results on other datasets are in Appendix B, where we observe similar trends. We make three observations. First, when the number of training steps is reduced, models fine-tuned

with dataset cartography clearly outperform those fine-tuned with ERM, even on ID validation data where ERM achieves better performance with more training steps. Second, dataset cartography often yields better results with fewer training steps than with more, suggesting a tendency of overfitting at the late training stage. Third, the efficiency gains hold consistently across different training durations. Our observations indicate that *training with data instances selected by DMs consistently achieves higher training speed* than ERM.

6 Our Approach: FTFT

Building on our insights presented in the previous sections, we propose a novel fine-tuning approach based on dataset cartography (Swayamdipta et al., 2020), **Fine-Tuning by transFerring Training dynamics (FTFT)**. Compared with the original dataset cartography approach, FTFT integrates two crucial improvements: (1) using more efficient reference models, which not only result in comparable DMs but also enhance efficiency, and (2) implementing aggressive early stopping during the main model training. Such aggressive early stopping is possible because models trained with data instances selected using DMs already achieve strong performance with substantially fewer training steps (§5). In summary, FTFT consists of three steps: (1) train an efficient reference model on the full dataset, (2) compute the ambiguity of each instance based on the standard deviation of the correct class probability across epochs, and (3) select the top $q\%$ most ambiguous instances to retrain a larger main model, while applying aggressive early stopping.

We show our results on NLI in Table 1 (FTFT) and on HSD in Table 5 (Appendix B). We use an early stopping patience $k = 2$ (i.e. the number of checkpoints without improvement before stopping), based on the average dev set performance on the OOD datasets. We use a relatively small k, as stopping the training earlier will lead to higher efficiency. We also show the computational cost of each method (Compute), in which we have taken the training of both the reference model and the main model into account, including the k extra checkpoints after the best performance is achieved.

We have three key observations. First, FTFT achieves consistent robustness improvements over ERM, indicated by its strong performance on most OOD datasets (note again that the hyper-parameters are optimized for ERM). We also include the re-sults of ERM with early stopping (ERM(ES)) as a baseline to illustrate that our aggressive early stopping strategy is only effective when training with data instances selected by DMs: ERM(ES) achieves worse performance than ERM on all OOD datasets. Second, FTFT substantially improves the training efficiency of models. For example, on NLI, training DeBERTaV3$_{\text{Large}}$ using FTFT and DeBERTaV3$_{\text{Small}}$ as the reference model only costs 51.57% of ERM's training cost, which is equivalent to 25.78% of the training cost of the original dataset cartography approach. Third, our early stopping strategy is effectively aggressive. Specifically, all FTFT results in Table 1 are achieved **using less than 1/3 of the optimal training steps needed for ERM**.

7 Conclusion

Fine-tuned PLMs have shown to be vulnerable to OOD input. Although dataset cartography can improve model robustness (Swayamdipta et al., 2020; Sar-Shalom and Schwartz, 2023), it is computationally expensive. In this paper, we have presented FTFT, a novel approach for fine-tuning PLMs which yields both better efficiency and better robustness over ERM (§6). FTFT is built on dataset cartography, based on two observations: (1) reference model training dynamics are highly transferable across different model sizes (§4.1) and pretraining methods (§4.2), and (2) main models trained using data instances selected using DMs learn faster (i.e. they have substantially better performance with reduced number of training steps). We believe that FTFT will be an important tool for future researchers and practitioners to perform efficient model fine-tuning, especially in situations where robustness is essential.

8 Limitations

We identify three limitations from this work. These limitations open up promising directions for future research. First, we have observed that effective reference models identify more instances as easy. More controlled experiments are needed to build a detailed protocol for choosing efficient but strong enough reference models. Our results provide a good foundation for such further work. Second, we have empirically demonstrated the transferability of training dynamics to select training instances. Future studies are needed to build the theoretical foundations of both data cartography itself and the

feasibility of such transfers. Third, we have only developed FTFT for classification tasks. Future studies should extend FTFT to generation tasks, e.g., instruction following and language modeling.

Acknowledgments

References

Robert John Nicholas Baldock, Hartmut Maennel, and Behnam Neyshabur. 2021. Deep learning through the lens of example difficulty. In *Advances in Neural Information Processing Systems*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*.

Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. 2017. Active bias: Training more accurate neural networks by emphasizing high variance samples. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pretraining text encoders as discriminators rather than generators. In *ICLR*.

Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. 2020. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*.

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.

Lucas Dixon, John Li, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. 2018. Measuring and mitigating unintended bias in text classification. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '18, page 67–73, New York, NY, USA. Association for Computing Machinery.

Yupei Du and Dong Nguyen. 2023. Measuring the instability of fine-tuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6209–6230, Toronto, Canada. Association for Computational Linguistics.

Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems*, volume 33, pages 2881–2891. Curran Associates, Inc.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTav3: Improving deBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. In *The Eleventh International Conference on Learning Representations*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. 2020. End-to-end bias mitigation by modelling biases in corpora. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8706–8716, Online. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu,

Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. 2021. Just train twice: Improving group robustness without training group information. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6781–6792. PMLR.

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*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

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

Marius Mosbach, Tiago Pimentel, Shauli Ravfogel, Dietrich Klakow, and Yanai Elazar. 2023. Few-shot fine-tuning vs. in-context learning: A fair comparison and evaluation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12284–12314, Toronto, Canada. Association for Computational Linguistics.

Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. 2020. Learning from failure: Training debiased classifier from biased classifier. In *Advances in Neural Information Processing Systems*.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Ji Ho Park, Jamin Shin, and Pascale Fung. 2018. Reducing gender bias in abusive language detection. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2799–2804, Brussels, Belgium. Association for Computational Linguistics.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. In *Advances in Neural Information Processing Systems*.

Alan Ramponi and Sara Tonelli. 2022. Features or spurious artifacts? data-centric baselines for fair and robust hate speech detection. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3027–3040, Seattle, United States. Association for Computational Linguistics.

Victor Sanh, Thomas Wolf, Yonatan Belinkov, and Alexander M Rush. 2021. Learning from others' mistakes: Avoiding dataset biases without modeling them. In *International Conference on Learning Representations*.

Aviad Sar-Shalom and Roy Schwartz. 2023. Curating datasets for better performance with example training dynamics. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10597–10608, Toronto, Canada. Association for Computational Linguistics.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari S. Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. In *Advances in Neural Information Processing Systems*.

Vladislav Sovrasov. 2018. ptflops: a flops counting tool for neural networks in pytorch framework.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online. Association for Computational Linguistics.

Kushal Tirumala, Aram H. Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. In *Advances in Neural Information Processing Systems*.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J. Gordon. 2019. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2020. Well-read students learn better: On the importance of pre-training compact models.

Prasetya Ajie Utama, Nafise Sadat Moosavi, and Iryna Gurevych. 2020. Mind the trade-off: Debiasing NLU models without degrading the in-distribution performance. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8717–8729, Online. Association for Computational Linguistics.

Bertie Vidgen, Dong Nguyen, Helen Margetts, Patricia Rossini, and Rebekah Tromble. 2021a. Introducing CAD: the contextual abuse dataset. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2289–2303, Online. Association for Computational Linguistics.

Bertie Vidgen, Tristan Thrush, Zeerak Waseem, and Douwe Kiela. 2021b. Learning from the worst: Dynamically generated datasets to improve online hate detection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1667–1682, Online. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi 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 M. 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.

Carole-Jean Wu, Ramya Raghavendra, Udit Gupta, Bilge Acun, Newsha Ardalani, Kiwan Maeng, Gloria Chang, Fiona Aga, Jinshi Huang, Charles Bai, Michael Gschwind, Anurag Gupta, Myle Ott, Anastasia Melnikov, Salvatore Candido, David Brooks, Geeta Chauhan, Benjamin Lee, Hsien-Hsin Lee, Bugra Akyildiz, Maximilian Balandat, Joe Spisak, Ravi Jain, Mike Rabbat, and Kim Hazelwood. 2022. Sustainable ai: Environmental implications, challenges and opportunities. In *Proceedings of Machine Learning and Systems*, volume 4, pages 795–813.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V. Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining. In *NeurIPS 2023*.

Michael Zhang, Nimit S Sohoni, Hongyang R Zhang, Chelsea Finn, and Christopher Re. 2022. Correct-n-contrast: a contrastive approach for improving robustness to spurious correlations. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 26484–26516. PMLR.

A Training Specifications

A.1 Experimental Setup

Optimization   For training all models, we use AdamW (Loshchilov and Hutter, 2019) as the optimizer with a batch size of 32. We also use a linear learning rate scheduler with 10% warmup. For fine-tuning the small and base versions of both DeBERTaV3 and ELECTRA, as well as TinyBERT, RoBERTa, and BERT, we use a learning rate of 2e-5, following the suggestions from the original papers. We use a smaller learning rate of 1e-5 for DeBERTa$_{Large}$ and ELECTRA$_{Large}$, because training larger models with lower learning rates are observed to be more stable (Mosbach et al., 2021; Du and Nguyen, 2023). However, a few failed training runs (i.e., runs where the training fails to converge and where the resulting model performs worse than the majority class baseline (Mosbach et al., 2021)) still occurred during the training of ELECTRA$_{Large}$. We excluded these runs from our results.

Number of Training Steps and Checkpointing Regarding the number of training steps, we perform a grid search for both DeBERTa$_{Large}$ and ELECTRA$_{Large}$ ERM models, spanning from one to five epochs, according to their average performance on the validation set of AdversarialNLI (NLI) and DynaHate (HSD). On NLI, the optimal length of training is four epochs (49088 steps) and five epochs (61360 steps) for DeBERTa$_{Large}$ and ELECTRA$_{Large}$. On HSD, the optimal length of training is three epochs (1620 steps) for both DeBERTa$_{Large}$ and ELECTRA$_{Large}$. For other PLMs, because we only use them as reference models, we do not perform this grid search, and use 61360 steps and 1620 steps for NLI and HSD. We perform checkpointing every 4090 steps and 180 steps on NLI and HSD, which is approximately the length of one epoch using the 33% selected data instances from DMs.

Software and Hardware   We use Python 3.9 and PyTorch 2.0 for all experiments. We also use HuggingFace Transformers 4.32 (Wolf et al., 2020), Accelerate 0.22, and Datasets 2.14 (Lhoest et al., 2021). All experiments are performed on one NVIDIA A100 GPU. Training all models (including hyper-parameter search) takes approximately 16 GPU days.

Model	Param Size	Compute
DeBERTaV3$_{Small}$	44.00M	14.47
DeBERTaV3$_{Base}$	86.00M	28.29
DeBERTaV3$_{Large}$	304.00M	100.00
ELECTRA$_{Small}$	14.00M	4.61
ELECTRA$_{Base}$	110.00M	36.18
ELECTRA$_{Large}$	335.00M	110.20
BERT$_{Large}$	345.00M	113.49
RoBERTa$_{Large}$	355.00M	116.78
TinyBERT	4.40M	1.45

Table 3: Model sizes and their estimated relative training cost in terms of FLOPs, estimated by following Kaplan et al. (2020).

A.2 Comparison of Training Costs

We estimate the training FLOPs by their models sizes (i.e. the number of parameters), following Kaplan et al. (2020) ($C \sim 6ND$, where C is the number of FLOPs, N is the number of parameters, and D is the dataset size). The results are in Table 3. We prefer theoretical estimation over practical measurement, because (1) the results are less influenced by noises, (2) it considers both forward and backward propagation, and (3) it has been shown effective and widely used (Kaplan et al., 2020; Hoffmann et al., 2022). In practice, we also measured the actual FLOPs usage in a forward run using Sovrasov (2018), and the results are very close to our estimation using model sizes.

B Additional Results

	NLI: MultiNLI				HSD: CAD			
	10%	25%	33%	50%	10%	25%	33%	50%
Model								
TinyBERT	80.01%	51.27%	38.34%	19.60%	81.89%	65.48%	58.20%	41.73%
ELECTRA$_{Small}$	80.10%	55.72%	47.40%	35.50%	82.97%	69.70%	61.92%	41.51%
ELECTRA$_{Base}$	84.57%	69.00%	62.84%	46.65%	86.10%	72.07%	63.41%	44.54%
DeBERTaV3$_{Small}$	82.80%	67.54%	61.72%	46.15%	85.44%	70.82%	62.83%	46.54%
DeBERTaV3$_{Base}$	85.08%	71.58%	64.06%	46.86%	85.49%	72.30%	63.64%	46.14%
BERT$_{Large}$	85.21%	70.08%	63.05%	47.21%	89.25%	72.38%	64.15%	47.07%
RoBERTa$_{Large}$	85.67%	71.43%	64.24%	47.20%	88.57%	73.21%	64.84%	47.25%

Table 4: Ratio of easy, i.e. neither ambiguous or hard-to-learn data, under different DM thresholds. Effective reference models identify larger easy data subsets, compared with the less effective reference models TinyBERT and ELECTRA$_{Small}$ This also implies that the hard-to-learn subsets and ambiguous subsets identified by effective reference models are very close to each other.
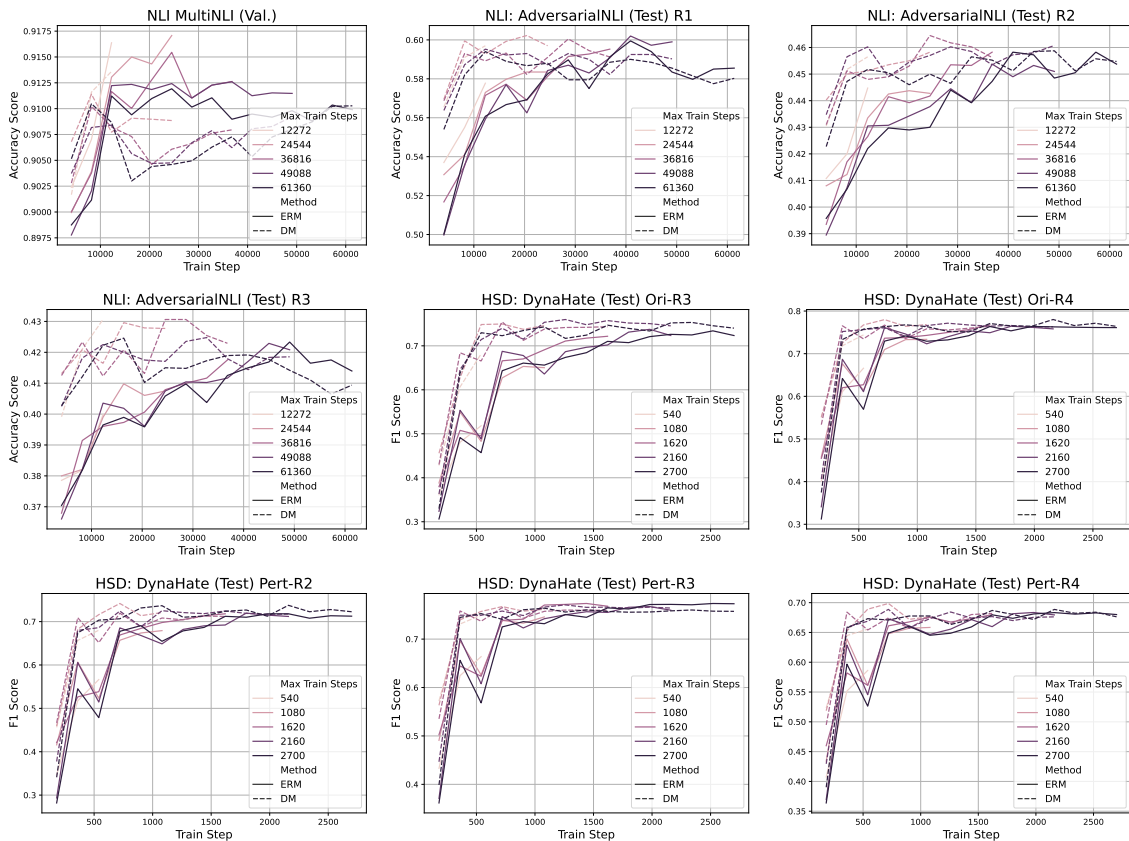
Figure 3: Performance when training the main model (DeBERTaV3$_{Large}$) using different numbers of training steps across different checkpoints. We experimented with different lengths of training (max training steps), and different methods (using ERM and DM). Training with data instances selected by DMs achieves consistent higher training speed than ERM: for datasets on which training with DM achieves either better or worse performance, models trained with DM outperform ERM with reduced training steps (i.e. the early stage of training, the leftmost part of the x-axis).

Method	Main Model	Ref. Model	Compute	CAD (Val.)	DynaHate (Test)					
				-	Ori-R2	Ori-R3	Ori-R4	Pert-R2	Pert-R3	Pert-R4
Baselines: ERM, ERM with Early Stopping, and Random DM										
ERM	DeBERTaV3$_{Small}$	-	14.47	$75.92_{1.33}$	$50.63_{2.65}$	$52.31_{2.60}$	$61.37_{1.44}$	$57.98_{2.42}$	$65.42_{0.92}$	$61.42_{0.50}$
ERM	DeBERTaV3$_{Base}$	-	28.29	$77.70_{1.11}$	$55.97_{0.75}$	$58.88_{2.84}$	$63.79_{1.84}$	$59.76_{0.94}$	$66.63_{1.53}$	$61.07_{0.95}$
ERM*	DeBERTaV3$_{Large}$	-	100.00	$80.95_{0.63}$	$77.30_{1.45}$	$72.17_{1.14}$	$76.75_{0.87}$	$71.80_{0.79}$	$76.79_{0.93}$	$67.99_{0.63}$
ERM(ES)	DeBERTaV3$_{Large}$	-	69.44	$79.01_{4.44}$	$66.86_{15.72}$	$64.37_{14.58}$	$72.36_{9.20}$	$68.22_{7.24}$	$73.65_{5.12}$	$66.44_{6.05}$
DM	DeBERTaV3$_{Large}$	Random	100.00	$76.02_{0.62}$	$64.53_{4.45}$	$63.44_{2.09}$	$73.13_{2.12}$	$64.23_{4.44}$	$71.84_{3.41}$	$64.03_{1.90}$
Training Dynamics Transferability: Across Different Model Sizes										
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Large}$	200.00	$80.52_{0.89}$	$79.57_{0.83}$	$74.34_{2.30}$	$76.47_{1.33}$	$71.19_{1.38}$	$75.96_{2.35}$	$68.19_{0.90}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Small}$	114.47	$81.27_{1.11}$	$76.46_{2.65}$	$74.71_{1.70}$	$77.00_{0.82}$	$72.82_{1.36}$	$76.99_{0.97}$	$68.79_{1.49}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Base}$	128.29	$80.63_{0.57}$	$78.98_{1.18}$	$74.46_{1.94}$	$75.87_{0.35}$	$73.37_{2.89}$	$77.26_{0.26}$	$67.57_{0.83}$
Training Dynamics Transferability: Across Different Pretraining Methods										
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Small}$	104.61	$76.69_{0.64}$	$73.12_{2.61}$	$71.64_{4.16}$	$76.91_{1.55}$	$70.42_{2.36}$	$76.68_{0.99}$	$68.77_{2.39}$
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Base}$	136.18	$80.38_{1.28}$	$78.55_{2.05}$	$76.75_{2.22}$	$77.55_{1.67}$	$73.86_{2.45}$	$77.11_{0.31}$	$69.35_{1.65}$
DM	DeBERTaV3$_{Large}$	RoBERTa$_{Large}$	216.78	$79.93_{0.94}$	$77.33_{1.71}$	$75.09_{1.68}$	$76.99_{1.53}$	$72.36_{0.54}$	$76.96_{1.19}$	$67.66_{1.84}$
DM	DeBERTaV3$_{Large}$	BERT$_{Large}$	213.49	$80.51_{0.92}$	$80.28_{3.19}$	$77.53_{2.00}$	$78.72_{1.55}$	$73.43_{2.24}$	$76.94_{1.39}$	$68.54_{1.32}$
FTFT: Efficient Reference Models + Aggressive Early Stopping										
FTFT	DeBERTaV3$_{Large}$	DeBERTaV3$_{Small}$	78.36	$80.20_{0.89}$	$78.13_{1.98}$	$74.85_{2.16}$	$77.59_{1.56}$	$72.44_{0.45}$	$77.44_{1.38}$	$69.28_{1.38}$
FTFT	DeBERTaV3$_{Large}$	DeBERTaV3$_{Base}$	106.07	$79.97_{1.17}$	$77.27_{3.93}$	$74.07_{1.93}$	$76.12_{0.86}$	$74.38_{2.68}$	$76.91_{1.85}$	$68.90_{1.07}$
FTFT	DeBERTaV3$_{Large}$	ELECTRA$_{Base}$	108.40	$79.40_{0.92}$	$80.10_{2.05}$	$76.26_{1.82}$	$79.07_{1.56}$	$75.69_{1.46}$	$77.93_{0.50}$	$69.44_{1.65}$

Table 5: Our results of DeBERTaV3 as the main model on HSD (measured by F1 scores), which consist of four parts: (1) Baselines: DeBERTaV3 of different sizes trained using ERM, DeBERTaV3$_{Large}$ trained using ERM with early stopping (ERM(ES)), and DeBERTaV3$_{Large}$ trained using random DM (random 33% of the training data); (2) Training dynamics transferability across different sizes: training DeBERTaV3$_{Large}$ as the main model, using DMs constructed by different sizes of DeBERTaV3 as reference models; (3) Training dynamics transferability across different pretraining methods: training DeBERTaV3$_{Large}$ as the main model, using DMs constructed by different pretraining methods as reference models, including ELECTRA$_{Small}$, ELECTRA$_{Base}$, TinyBERT, and RoBERTa; (4) FTFT: training DeBERTaV3$_{Large}$ using our approach FTFT, with DMs constructed by different reference models, as well as aggressive early stopping. Ori/Pert and R2–R4 in DynaHate refer to different rounds of collected **Original** and **Perturbed** data. Compute refers to the relative training computational cost compared to training DeBERTaV3$_{Large}$ using ERM. We observe that: (1) Training dynamics are transferrable across different sizes and pretraining methods, as constructing DMs using different reference models results in comparable performance; (2) FTFT achieves consistent robustness improvements over ERM, while maintaining or lowering the training cost. (3) FTFT enhances efficiency more when the optimal length of training is longer (ERM only trains 1.6k steps on CAD).

Method	Main Model	Ref. Model	Compute	CAD (Val.)	DynaHate (Test)					
					Ori-R2	Ori-R3	Ori-R4	Pert-R2	Pert-R3	Pert-R4
ERM Performance: Capabilities of Various Models										
ERM	TinyBERT	-	1.45	$63.85_{0.95}$	$36.32_{0.91}$	$41.68_{2.40}$	$44.93_{0.68}$	$31.99_{1.62}$	$44.28_{2.49}$	$43.12_{1.17}$
ERM	ELECTRA$_{Small}$	-	4.61	$71.55_{0.86}$	$47.36_{2.31}$	$47.83_{0.90}$	$55.23_{2.48}$	$48.30_{2.28}$	$57.18_{2.28}$	$55.37_{1.95}$
ERM	ELECTRA$_{Base}$	-	36.18	$76.62_{1.23}$	$59.43_{2.73}$	$57.86_{2.35}$	$65.75_{1.59}$	$57.49_{3.21}$	$65.15_{0.56}$	$63.48_{1.33}$
ERM	DeBERTaV3$_{Small}$	-	14.47	$75.92_{1.20}$	$50.63_{2.40}$	$52.31_{2.35}$	$61.37_{1.30}$	$57.98_{2.19}$	$65.42_{0.83}$	$61.42_{0.45}$
ERM	DeBERTaV3$_{Base}$	-	28.29	$77.70_{1.01}$	$55.97_{0.68}$	$58.88_{2.57}$	$63.79_{1.67}$	$59.76_{0.85}$	$66.63_{1.39}$	$61.07_{0.86}$
ERM	BERT$_{Large}$	-	113.49	$77.57_{0.77}$	$55.80_{3.04}$	$61.62_{3.00}$	$64.60_{3.53}$	$61.82_{0.78}$	$64.69_{1.26}$	$65.00_{2.48}$
ERM	RoBERTa$_{Large}$	-	116.78	$78.92_{0.94}$	$64.68_{3.82}$	$65.95_{1.10}$	$68.65_{1.55}$	$63.73_{0.55}$	$70.39_{0.46}$	$64.27_{1.50}$
When Do Transfers Fail: Using Reference Models of Different Capabilities										
DM	DeBERTaV3$_{Large}$	TinyBERT	101.45	$77.97_{1.07}$	$68.51_{2.47}$	$68.89_{1.73}$	$73.99_{1.23}$	$65.60_{3.96}$	$73.22_{2.16}$	$66.36_{0.94}$
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Small}$	104.61	$76.69_{0.58}$	$73.12_{2.36}$	$71.63_{3.76}$	$76.91_{1.41}$	$70.42_{2.13}$	$76.68_{0.89}$	$68.77_{2.16}$
DM	DeBERTaV3$_{Large}$	ELECTRA$_{Base}$	136.18	$80.38_{1.16}$	$78.55_{1.86}$	$76.75_{2.01}$	$77.55_{1.51}$	$73.86_{2.21}$	$77.11_{0.28}$	$69.35_{1.49}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Small}$	114.47	$81.27_{1.00}$	$76.46_{2.39}$	$74.71_{1.54}$	$77.00_{0.74}$	$72.82_{1.23}$	$76.99_{0.87}$	$68.79_{1.35}$
DM	DeBERTaV3$_{Large}$	DeBERTaV3$_{Base}$	128.29	$80.63_{0.52}$	$78.98_{1.07}$	$74.46_{1.75}$	$75.87_{0.31}$	$73.37_{2.61}$	$77.25_{0.24}$	$67.57_{0.75}$
DM	DeBERTaV3$_{Large}$	BERT$_{Large}$	213.49	$80.51_{0.92}$	$80.28_{3.19}$	$77.53_{2.00}$	$78.72_{1.55}$	$73.43_{2.24}$	$76.94_{1.39}$	$68.54_{1.32}$
DM	DeBERTaV3$_{Large}$	RoBERTa$_{Large}$	216.78	$79.93_{0.85}$	$77.33_{1.54}$	$75.09_{1.52}$	$76.99_{1.38}$	$72.36_{0.49}$	$76.96_{1.07}$	$67.66_{1.67}$
DM	ELECTRA$_{Large}$	TinyBERT	111.64	$76.64_{0.58}$	$67.59_{1.86}$	$65.36_{1.97}$	$71.35_{1.84}$	$64.85_{2.66}$	$70.37_{0.74}$	$67.77_{1.57}$
DM	ELECTRA$_{Large}$	ELECTRA$_{Small}$	114.80	$71.40_{11.28}$	$57.75_{16.82}$	$57.93_{14.85}$	$63.25_{17.15}$	$57.05_{17.35}$	$63.60_{14.16}$	$62.05_{12.46}$
DM	ELECTRA$_{Large}$	ELECTRA$_{Base}$	146.38	$78.61_{0.31}$	$76.33_{1.04}$	$69.80_{2.10}$	$74.23_{1.65}$	$69.12_{0.95}$	$71.72_{1.39}$	$69.23_{1.22}$
DM	ELECTRA$_{Large}$	DeBERTaV3$_{Small}$	124.67	$78.37_{0.81}$	$76.39_{0.75}$	$68.51_{0.77}$	$72.12_{1.87}$	$68.81_{0.80}$	$72.58_{0.97}$	$68.83_{0.99}$
DM	ELECTRA$_{Large}$	DeBERTaV3$_{Base}$	138.49	$74.87_{3.75}$	$59.49_{20.98}$	$58.43_{13.10}$	$65.65_{9.34}$	$57.63_{14.30}$	$65.20_{9.59}$	$60.47_{10.88}$
DM	ELECTRA$_{Large}$	BERT$_{Large}$	223.68	$78.38_{1.27}$	$76.69_{3.87}$	$71.80_{1.82}$	$74.63_{1.59}$	$71.62_{1.36}$	$73.23_{1.37}$	$69.09_{0.57}$
DM	ELECTRA$_{Large}$	RoBERTa$_{Large}$	226.97	$76.64_{3.91}$	$66.46_{10.28}$	$65.42_{1.06}$	$72.47_{2.36}$	$66.79_{4.70}$	$71.58_{1.91}$	$66.45_{1.27}$
DM	DeBERTaV3$_{Base}$	TinyBERT	29.74	$75.00_{0.01}$	$53.00_{0.06}$	$58.00_{0.06}$	$63.00_{0.04}$	$57.00_{0.05}$	$66.00_{0.03}$	$60.00_{0.04}$
DM	DeBERTaV3$_{Base}$	DeBERTaV3$_{Small}$	42.76	$77.00_{0.02}$	$64.00_{0.03}$	$67.00_{0.02}$	$70.00_{0.02}$	$64.00_{0.03}$	$69.00_{0.03}$	$63.00_{0.02}$

Table 6: We use reference models of different capabilities (measured by their ERM F1 scores) to construct DMs, and use them to train main models. The gray-shaded rows are the (1) main models in unsuccessful transfers and (2) their corresponding reference models. The orange-shaded rows are the rows with very high standard deviations: we observe that **fine-tuning ELECTRA$_{Large}$ on small datasets like CAD is very unstable, and often produces failed runs.** Following Mosbach et al. (2021) we removed the runs with ID performance worse than the majority classifier: however, there are still some runs with slightly better ID performance than the majority classifier, but with diverged loss or fluctuating loss after the first a few training steps (i.e. the loss curve going up, or the loss curve being almost flat), and are significantly worse than the other runs. We therefore **exclude these runs in our analyses in §4.3.** These "almost failed runs" cause the high standard deviations in the orange-shaded rows. Successful transfer requires the reference model to be reasonably strong: reference models with clearly worse ID performance lead to degraded OOD performance for the main models.