Data Selection for Fine-tuning Large Language Models Using Transferred Shapley Values

Stephanie Schoch **Ritwick Mishra** **Yangfeng Ji**
Department of Computer Science
University of Virginia
Charlottesville, VA 22904
{sns2gr,mbc7bu,yangfeng}@virginia.edu

Abstract

Although Shapley values have been shown to be highly effective for identifying harmful training instances, dataset size and model complexity constraints limit the ability to apply Shapley-based data valuation to fine-tuning large pre-trained language models. To address this, we propose TS-DSHAPLEY, an algorithm that reduces computational cost of Shapley-based data valuation through: 1) an efficient sampling-based method that aggregates Shapley values computed from subsets for valuation of the entire training set, and 2) a value transfer method that leverages value information extracted from a simple classifier trained using representations from the target language model. Our experiments applying TS-DSHAPLEY to select data for fine-tuning BERT-based language models on benchmark natural language understanding (NLU) datasets show that TS-DSHAPLEY outperforms existing data selection methods. Further, TS-DSHAPLEY can filter fine-tuning data to increase language model performance compared to training with the full fine-tuning dataset.

1 Introduction

Large language models (LMs) have achieved state-of-the-art performance on many natural language processing (NLP) tasks (Radford et al., 2019; Brown et al., 2020; Sanh et al., 2022). To adapt these models to new datasets and tasks, the standard approach is to fine-tune a pre-trained LM on a targeted downstream task. This allows the pre-trained general linguistic knowledge to be leveraged while fine-tuning to learn the task-specific information. However, during fine-tuning, pre-trained LMs are prone to significant performance degradation in the presence of noisy data (Srivastava et al., 2020). This effect may be further amplified when noisy or otherwise harmful instances are highly influential to the model parameters (Koh and Liang, 2017). As a result, it is important to identify harmful instances in the fine-tuning data that may obfuscate the task information and degrade performance.

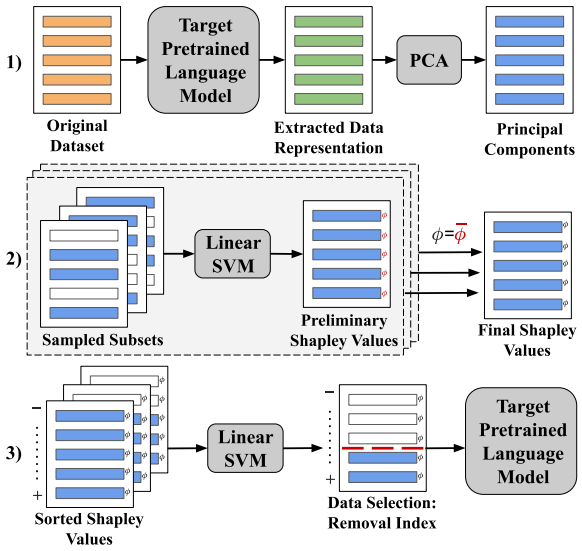

Figure 1: An overview of TS-DSHAPLEY: 1) Process the data using the target LM; 2) Compute *sampling* chains using a subset of the training set and aggregate the resulting Shapley values; and 3) *Transfer* the estimated data value information for use with the target LM by estimating the optimal low value data removal index.

To automatically identify harmful data, prior works have used training dynamics (Swayamdipta et al., 2020) and estimation of marginal contributions via leave-one-out retraining (Cook, 1977) or influence functions (Koh and Liang, 2017). Shapley values, which satisfy certain desirable fairness guarantees, have also recently been adopted from cooperative game theory to measure datum contributions, where a data point's Shapley value is the average marginal contribution to every possible data subset (Ghorbani and Zou, 2019).

In practice, Shapley-based data values are approximated using various techniques (Ghorbani and Zou, 2019; Jia et al., 2019b, 2021; Kwon and Zou, 2022; Schoch et al., 2022), as exact Shapley value computation over a dataset would require *exhaustively retraining the model* for every datum on

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics - Student Research Workshop, pages 266–275
July 10-12, 2023

every possible subset (i.e. *exponential complexity with respect to the number of data points*). However, many of the existing approximation methods still exhibit a computational bottleneck when considering datasets and models at scale (e.g. datasets larger than 5K instances). This, in turn, directly limits the application of Shapley-based data valuation to state-of-the-art LMs and many NLP datasets.

To address the challenges posed by 1) the *model constraint* (the model retraining requirement) and 2) the *dataset constraint* (the time-complexity/dataset size relation), we propose Transferred Sampling Data Shapley (TS-DSHAPLEY), an algorithm that utilizes two novel components that directly address each constraint. Specifically, to address the model constraint, we propose to compute Shapley-based data values using a simple, linear model that is trained on the learned representation from the target LM. Additionally, to address the dataset constraint, we propose a sampling-based method that computes Shapley values on data subsets and aggregates them for valuation of the entire training set.

Our contributions are as follows: 1) we propose a sampling-based data Shapley computation method and demonstrate its efficacy empirically using as little as 2% of the original training data; 2) we propose the use of a simple linear classifier with a target model's pre-trained representation and demonstrate empirically the performance gains achieved over alternate pre-trained embeddings; and 3) we show the efficacy of Shapley-based data valuation and selection methods on benchmark NLU tasks using fine-tuned large LMs.[1]

2 Related Work

While Shapley values are often applied in a post hoc manner following model training (Ghorbani and Zou, 2019; Kwon and Zou, 2022; Jia et al., 2019a,b, 2021; Schoch et al., 2022), the demonstrated efficacy makes it a natural extension to apply such methods for data selection *prior to* training. To this end, Shapley values have been used for evaluating data for transfer learning (Parvez and Chang, 2021) and in active learning (Ghorbani et al., 2021).

Further, although Shapley-based data values have primarily been considered model-specific, in practice, a subset of training instances that may harm performance may be mislabeled (Koh and Liang, 2017; Swayamdipta et al., 2020; Ghorbani and Zou, 2019) or exhibit spelling mistakes or grammatical errors (Sun et al., 2020; Srivastava et al., 2020), which should be intrinsic to the dataset. Prior works have demonstrated the transferability of Shapley-based data values across various classifier architectures (Schoch et al., 2022) and have demonstrated the efficacy of surrogate KNN classifiers using pre-trained embeddings (Jia et al., 2021). Notably, our work differs in that we utilize the pre-trained embeddings extracted from the target LM and avoid the k-nearest neighbor assumption that training data far from a test datum do not contribute to its prediction (Jia et al., 2019a).

3 Method

Let $D = \{(x_i, y_i)\}_{i=1}^n$ denote a training set containing n training instances. For each training instance i, the Shapley value ϕ_i is defined as the average marginal contribution of i to every possible subset $S \subseteq D$ that contains this instance (Ghorbani and Zou, 2019): $\phi_i = \sum_{S \subseteq D; i \in S} \frac{1}{\binom{n-1}{|S \setminus \{i\}|}} \{v_{\mathcal{A}}(S) - v_{\mathcal{A}}(S \setminus \{i\})\}$ where $v_{\mathcal{A}}(S)$ is a value function, typically defined as the development accuracy of model $\mathcal{A}$ trained on S. The challenge of calculating ϕ_i is two-fold: the exponential complexity of all possible subsets $S \subseteq D$ and the computational cost of training $\mathcal{A}$ on each S and $S \setminus \{i\}$. While Shapley-based data values are approximated in practice, most existing approximation methods are not efficient enough for large scale learning problems.

3.1 TS-DSHAPLEY

Let $\mathcal{A}_{tgt}$ be the target classifier (i.e. large LM) that we want to fine-tune on a subset of D. To reduce computational cost, we propose to (1) use a linear classifier $\mathcal{A}_{src}$ as the proxy of $\mathcal{A}_{tgt}$ for data valuation; (2) use multi-chain Monte Carlo sampling to compute Shapley values on different subsets of D. For faithful data valuation, we further propose to train $\mathcal{A}_{src}$ on the data representations extracted from $\mathcal{A}_{tgt}$.

Representation Extraction. We extract the representations from the penultimate layer of the pre-trained LM $\mathcal{A}_{tgt}$ as the inputs for training $\mathcal{A}_{src}$. Note that training $\mathcal{A}_{src}$ in this way is equivalent to fixing the LM and only fine-tuning the last classification layer. To further remove the redundancy in

[1]Code is available at https://github.com/stephanieschoch/ts-dshapley

the representations and reduce computational cost, we follow prior work by performing PCA on the collection of representations and selecting the first 32 principal components (Ghorbani and Zou, 2019; Kwon and Zou, 2022; Schoch et al., 2022).

Sampling Data Shapley. Instead of directly estimating Shapley-based data values via Monte Carlo sampling on the whole training set, our approach performs Monte Carlo sampling on subsets of the data, which we refer to as *sampling chains*. Within a single sampling chain c, we sample a subset of training instances S_t, estimate their contributions, and repeat T times. The contribution of each instance in S_t is calculated by removing one instance at a time in a random order. For example, the contribution of the first randomly removed instance i is $c_{S_t}(i) = v_{\mathcal{A}_{src}}(S_t) - v_{\mathcal{A}_{src}}(S_t \backslash \{i\})$, the contribution of the second randomly removed instance k is $c_{S_t}(k) = v_{\mathcal{A}_{src}}(S_t \backslash \{i\}) - v_{\mathcal{A}_{src}}(S_t \backslash \{i, k\})$, and so on. On the other hand, if an instance i is not in S_t, $c_{S_t}(i) = 0$.

After T times, the Shapley value of instance i is approximated as $\phi_i \approx \frac{1}{T} \sum_{S_t} c_{S_t}(i)$. To balance the computational efficiency and approximation, we empirically define a range of the size $|S_t| \in [\frac{s}{2}, s]$, with subset size s as the sampling upper bound.

Computation can be further sped up with multiple Monte Carlo sampling chains $S_t^{(c)}, c \in \{1, \ldots, J\}$. The corresponding value approximation is defined as $\phi_i = \frac{1}{J} \sum_c \frac{1}{T} \sum_{S_t^{(c)}} c_{S_t^{(c)}}(i)$. As each chain can be computed independently, the efficiency can be boosted with parallel computing. This novel idea of multi-chain sampling serves as the core of TS-DSHAPLEY and significantly speeds up computation, in practice working with a simple model $\mathcal{A}_{src}$.

Data Selection with TS-DSHAPLEY Values. To identify harmful data points, we use the data removal strategy of Ghorbani and Zou (2019) on $\mathcal{A}_{src}$ and transfer the selection outcome to the target model $\mathcal{A}_{tgt}$. Specifically, we gradually remove training instances from the lowest estimated contribution value to the highest estimated contribution value. Following each removal, we retrain A_{src} and evaluate predictive performance on the held-out development data. As a result, this removal procedure will identify a optimal subset S_{opt} that gives the best predictive performance on A_{src}. With the assumption of data value transferability (Schoch et al., 2022), we expect that $\mathcal{A}_{tgt}$ trained on S_{opt} will give no worse, and likely better performance, than $\mathcal{A}_{tgt}$ trained on D. While this data removal strategy is proposed in prior work (Ghorbani and Zou, 2019), the data selection use case is novel in NLP.

4 Experiments

4.1 Experiment Setup

Pre-trained Large Language Models. We utilize two transformer-based large LMs for which traditional Shapley-based data value computation would be intractable: RoBERTa-base (Liu et al., 2019, 125M parameters) and DistilBERT (Sanh et al., 2019, 66M parameters).

Datasets. We select one GLUE benchmark (Wang et al., 2019) dataset from each task category: SST-2 (Socher et al., 2013), QQP (Iyer et al., 2017), and RTE (Dagan et al., 2006), representing Single-Sentence Tasks, Similarity and Paraphrase Tasks, and Inference Tasks, respectively. Additional dataset details are reported in Appendix A. Notably, we select datasets of varied sizes to reflect diverse sampling subset to training set size ratios.

Data Selection Baselines. We compare against performance when training on the full data subset as well as three selection baselines: leave-one-out (LOO) (Cook, 1977), KNN-shapley (KNN) (Jia et al., 2019a, 2021), and random sampling. For LOO, we use the same classifier architecture as with TS-DSHAPLEY to compute value estimates. For both LOO and KNN, we reduce the dataset using the data removal procedure defined in section 3. Finally, for random sampling, we remove a random sample of data points equal to the number of points removed via TS-DSHAPLEY.

4.2 Data Selection Experiment

To test the efficacy of using TS-DSHAPLEY to select data for fine-tuning large LMs, we compute data values using each method and perform the data removal procedure described in section 3. Specifically, we remove the lowest value data points preceding the data removal step that achieved the highest development accuracy using A_{src}. For TS-DSHAPLEY, we vary the subset size and number of chains based on dataset size, using subset size $= 6.7k(10\%), 7.28k(2\%), 374(15\%)$ and number of chains $= 25, 10, 25$ for SST-2, QQP, and RTE,

Method Category	Method	RoBERTa			DistilBERT		
		SST-2	QQP	RTE	SST-2	QQP	RTE
Full Training Set	Liu et al. (2019)	0.948	0.919	0.787	–	–	–
	Sanh et al. (2019)	–	–	–	0.913	0.885	0.599
	Full Dataset	0.950	0.917	0.788	0.908	0.905	0.618
Data Selection Baselines	Leave-One-Out	0.947	–	0.784	0.912	–	0.614
	KNN Shapley	0.946	0.916	0.781	0.911	0.905	0.622
	Random	0.947	0.917	0.684	0.911	0.905	0.589
Our Method	TS-DSHAPLEY	**0.953**	**0.919**	**0.801**	**0.915**	**0.907**	**0.652**

Table 1: Predictive accuracy when selecting data using each valuation method. Results reflect the mean of five trials. We do not report LOO as a baseline for QQP due to computational intractability.

respectively. Additional training and hyperparameter details, including details of a limited hyperparameter sweep, can be found in Appendix A.

Results Results are shown in Table 1. TS-DSHAPLEY consistently outperforms baseline selection methods as well as performance using the full fine-tuning dataset. Notably, data selection using TS-DSHAPLEY resulted in performance improvements of up to 1.3% and 3.4% for RoBERTa and DistilBERT, respectively, over the predictive performance when training using the full fine-tuning dataset. These results indicate TS-DSHAPLEY successfully identifies data points that harm model performance. As an additional analysis, for the RTE dataset we show the location of harmful points identified by TS-DSHAPLEY on a data map (Swayamdipta et al., 2020) in Appendix B.

4.3 Sampling Hyperparameter Analysis

TS-DSHAPLEY exhibited good performance for data selection across various subset sizes and numbers of chains. For example, on QQP TS-DSHAPLEY outperformed the full dataset and baseline methods when using a subset of just 2% of the training set. To better understand the impact of different parameter values, we utilize a parameter value grid on the RTE dataset and re-compute TS-DSHAPLEY. Specifically, using the best hyperparameters from subsection 4.2 (see Appendix A), we evaluate performance of RoBERTa and DistilBERT using a parameter sweep of subset size as a percentage of the total training set size, subset size $\in \{1, 2, 5, 10, 15\}\%$, and number of chains $\in \{2, 5, 10, 15\}$ and report the Pearson's correlation between each parameter and performance.

Results. All correlations are reported in Appendix B and summarized here. When subset size $> 2\%$, both models demonstrate a high positive correlation between number of chains and performance. For example, when using 15% of the training data, RoBERTa on RTE had a correlation of 0.94. Across the different number of chains, however, there was no consistent pattern of correlation between subset size and performance. This indicates that increasing number of chains (which can be computed in-parallel) may be of more benefit compared to increasing sampling subset size.

Model	Embeddings	SST-2	QQP	RTE
RoBERTa	RoBERTa	**0.953**	**0.919**	**0.801**
	DistilBERT	0.951	0.906	0.762
	GloVe	0.948	0.908	0.767
DistilBERT	DistilBERT	**0.915**	**0.907**	**0.652**
	RoBERTa	0.906	0.903	0.623
	GloVe	0.909	0.903	0.632

Table 2: Predictive accuracy using TS-DSHAPLEY with different word embeddings.

4.4 Effect of Different Embeddings

To test the efficacy of computing TS-DSHAPLEY using the extracted representations from the target LM, we perform an experiment where we use the removal indices computed with 1) the representation from a different language model (e.g. removing indices for fine-tuning RoBERTa using the optimal removal index identified using DistilBERT data representations), and 2) GloVe pre-trained word embeddings (Pennington et al., 2014), as a third-party representation repository.

Results. As shown in Table 2, while alternate embeddings can still lead to improvements over the full data, using the representation from the target LM is beneficial and consistently outperforms other embeddings. The results suggest that low value data is likely a combination of (i) inherently noisy

data (e.g. mislabeled instances) and (ii) instances that are harmful to specific models due to different model architectures and pre-training strategies.

5 Conclusion

In this work, we propose TS-DSHAPLEY to address the model and dataset constraints that currently contribute to a computational bottleneck when computing Shapley-based data value estimates.

Limitations

While we demonstrate the efficacy of TS-DSHAPLEY empirically, the current work is limited in terms of theoretical analysis. For example, while we have good empirical performance with a linear SVM, additional analysis could determine if there are optimal ways to select an alternative simple model architecture for the source classifier depending on the target classifier or dataset. Additionally, while we found a strong correlation between number of sampling chains and performance when the subset size was $> 2\%$ of the training data size, the lower subset size threshold to observe this correlation may be dataset dependent, which additional analysis could address.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

R Dennis Cook. 1977. Detection of influential observation in linear regression. *Technometrics*, 19(1):15–18.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine learning challenges. evaluating predictive uncertainty, visual object classification, and recognising tectual entailment*, pages 177–190. Springer.

Amirata Ghorbani and James Zou. 2019. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, pages 2242–2251. PMLR.

Amirata Ghorbani, James Zou, and Andre Esteva. 2021. Data shapley valuation for efficient batch active learning. *arXiv preprint arXiv:2104.08312*.

Shankar Iyer, Nikhil Dandekar, and Kornel Csernai. 2017. First quora dataset release: Question pairs.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas Spanos, and Dawn Song. 2019a. Efficient task-specific data valuation for nearest neighbor algorithms. *Proceedings of the VLDB Endowment*, 12(11):1610–1623.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. 2019b. Towards efficient data valuation based on the shapley value. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 1167–1176. PMLR.

Ruoxi Jia, Fan Wu, Xuehui Sun, Jiacen Xu, David Dao, Bhavya Kailkhura, Ce Zhang, Bo Li, and Dawn Song. 2021. Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8239–8247.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Yongchan Kwon and James Zou. 2022. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022*.

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*.

Md Rizwan Parvez and Kai-Wei Chang. 2021. Evaluating the values of sources in transfer learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5084–5116.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja,

et al. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations*.

Stephanie Schoch, Haifeng Xu, and Yangfeng Ji. 2022. Cs-shapley: Class-wise shapley values for data valuation in classification. In *Advances in Neural Information Processing Systems*.

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

Ankit Srivastava, Piyush Makhija, and Anuj Gupta. 2020. Noisy text data: Achilles' heel of bert. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 16–21.

Lichao Sun, Kazuma Hashimoto, Wenpeng Yin, Akari Asai, Jia Li, Philip Yu, and Caiming Xiong. 2020. Adv-bert: Bert is not robust on misspellings! generating nature adversarial samples on bert. *arXiv preprint arXiv:2003.04985*.

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.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In the Proceedings of ICLR.

A Additional Experiment Details

In this section, we include additional experiment setup details.

A.1 Datasets

Dataset statistics are provided in Table 3, with further description provided below.

SST-2: Stanford Sentiment Treebank (Socher et al., 2013) is a collection of English movie reviews with human annotations of their sentiment. The model is tasked with predicting a review's sentiment as positive or negative.

QQP: Quora Question Pairs (Iyer et al., 2017) is a collection of English question pairs from the website Quora where the task is to determine if a pair of questions are similar in meaning.

RTE: Recognizing Textual Entailment (Dagan et al., 2006) combines several English datasets from annual textual entailment challenges, where the task is to predict if the *text* entails the *hypothesis* or not.

A.2 Hyperparameters

For each experiment, we consider a limited hyperparameter sweep for each model, selection method, and task, with batch size $\in \{16, 32\}$ and learning rate $\in \{10^{-5}, 3 \times 10^{-5}\}$. The rest of the hyperparameters are kept consistent across experiment conditions. We report the mean development set accuracy from five random initializations for which we fine-tune for 10 epochs and select the model checkpoint with the highest development set accuracy. Results from each hyperparameter sweep are reported in Table 4 and Table 5.

B Additional Results

B.1 Additional Data Selection Analysis

While we compare directly with baseline selection methods that directly measure estimated data contribution, we perform an additional analysis by comparing the indices removed with TS-DSHAPLEY with the mapped training dynamics using data maps (Swayamdipta et al., 2020). Specifically, we first plot the data map for RoBERTa trained on RTE using the same hyperparameters as in subsection 4.2. Then, we plot the same data map showing only the data points that were identified by TS-DSHAPLEY to be harmful, i.e. removed from the fine-tuning training data. These are shown in Figure 2 and Figure 3, respectively.

We observe that a handful of instances in the hard-to-learn region (identified by Swayamdipta et al. (2020) to contain some mislabeled examples) were removed, as well as a small number of instances in the ambiguous region. Interestingly though, we observe that 1) most of the data points in RTE belonged to the easy-to-learn region, and 2) a cluster of easy-to-learn points were removed. Swayamdipta et al. (2020) found that too many easy-to-learn instances could decrease both in-distribution and out-of-distribution performance and noted that determining how to select an optimal balance of easy-to-learn and ambiguous examples, particularly in low data settings, was an open problem. As TS-DSHAPLEY achieved a performance gain over the full dataset performance, these results suggest that TS-DSHAPLEY may be effective to potentially determine an optimal balance and address this problem. We leave further analysis of this to future work.

B.2 Sampling Hyperparameter Analysis.

Pearson's correlation coefficients for the sampling parameter analysis in section 4 are reported in Table 6 and Table 7, where each result represents the mean of five sampling and chain computation trials.

Dataset	GLUE Task Category	Task	Metric	Data Split	
				Train	Dev
SST-2	Single Sentence Tasks	Sentiment	Acc.	67k	1.8k
QQP	Similarity and Paraphrase Tasks	Paraphrase	Acc./F1	364k	40.4k
RTE	Inference Tasks	NLI	Acc.	2.5k	277

Table 3: Statistics for each dataset. We use the train and development data splits as GLUE tasks have held out test set labels.

Model	Method	SST-2		QQP		RTE	
		BS	LR	BS	LR	BS	LR
RoBERTa	Full Dataset	16	10^{-5}	32	3×10^{-5}	16	3×10^{-5}
	Leave-One-Out	32	10^{-5}	–	–	16	3×10^{-5}
	KNN Shapley	16	10^{-5}	32	3×10^{-5}	16	3×10^{-5}
	Random	32	3×10^{-5}	32	3×10^{-5}	16	3×10^{-5}
	TS-DSHAPLEY	32	10^{-5}	32	3×10^{-5}	16	3×10^{-5}
DistilBERT	Full Dataset	16	10^{-5}	32	3×10^{-5}	32	3×10^{-5}
	Leave-One-Out	32	10^{-5}	–	–	16	10^{-5}
	KNN Shapley	16	10^{-5}	32	3×10^{-5}	16	10^{-5}
	Random	32	3×10^{-5}	16	3×10^{-5}	16	3×10^{-5}
	TS-DSHAPLEY	16	3×10^{-5}	16	10^{-5}	16	3×10^{-5}

Table 4: Batch size (BS) and learning rate (LR) for the data selection experiment based on the hyperparameter sweep defined in section 4.

Model	Embeddings	SST-2		QQP		RTE	
		BS	LR	BS	LR	BS	LR
RoBERTa	RoBERTa	32	10^{-5}	32	3×10^{-5}	16	3×10^{-5}
	DistilBERT	16	10^{-5}	32	10^{-5}	16	3×10^{-5}
	GloVe	16	3×10^{-5}	32	3×10^{-5}	32	3×10^{-5}
DistilBERT	DistilBERT	16	10^{-5}	16	10^{-5}	16	3×10^{-5}
	RoBERTa	32	10^{-5}	32	10^{-5}	32	10^{-5}
	GloVe	32	10^{-5}	32	3×10^{-5}	32	3×10^{-5}

Table 5: Batch size (BS) and learning rate (LR) for the embeddings switch experiment based on the hyperparameter sweep defined in section 4.

Model	Subset Size (%, #)				
	1 (25)	2 (50)	5 (125)	10 (249)	15 (374)
RoBERTa	0.119	0.013	0.892	0.929	0.942
DistilBERT	0.240	0.104	0.613	0.776	0.714

Table 6: Correlations between number of chains and performance for each subset size on the RTE dataset.

Model	Number of Sampling Chains					
	2	5	10	15	20	25
RoBERTa	-0.463	0.127	-0.474	0.013	0.472	0.763
DistilBERT	0.027	-0.034	0.530	0.447	0.737	0.692

Table 7: Correlations between subset size and performance for each number of sampling chains on the RTE dataset.

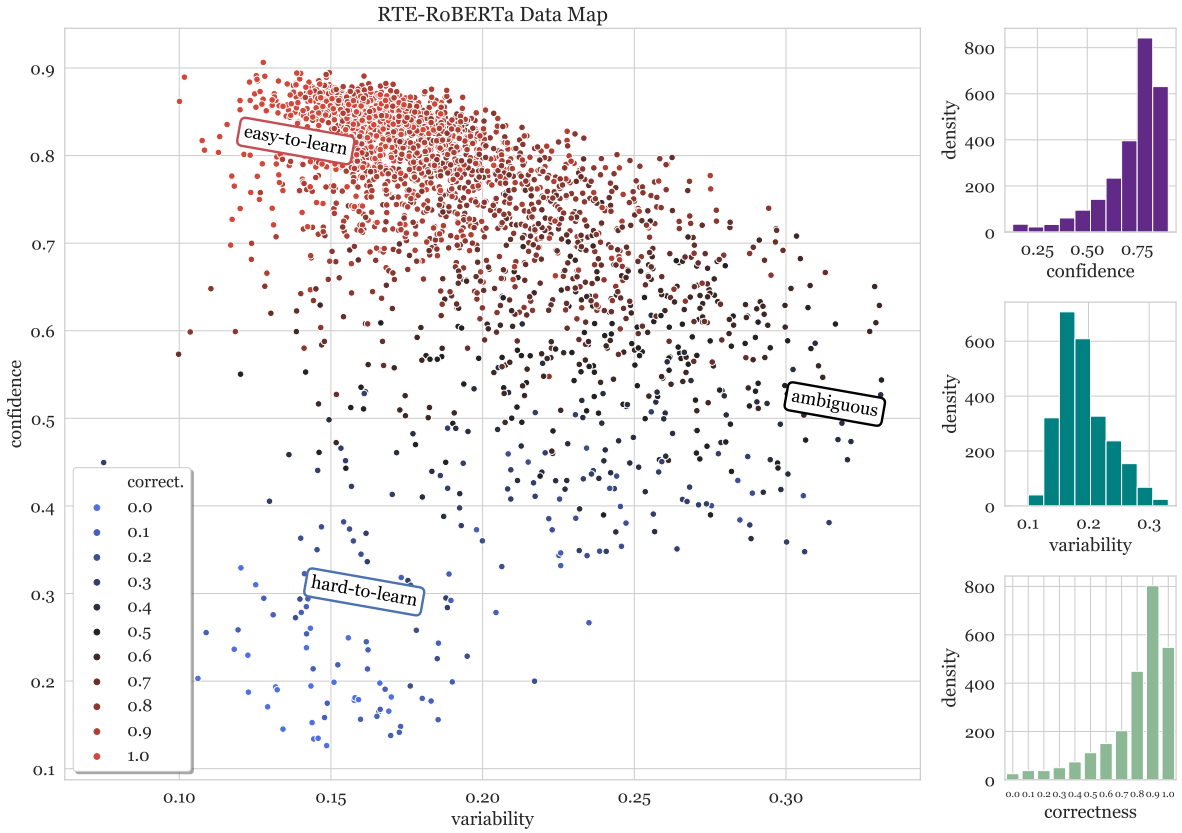

Figure 2: Data map for RoBERTa trained on the RTE dataset.

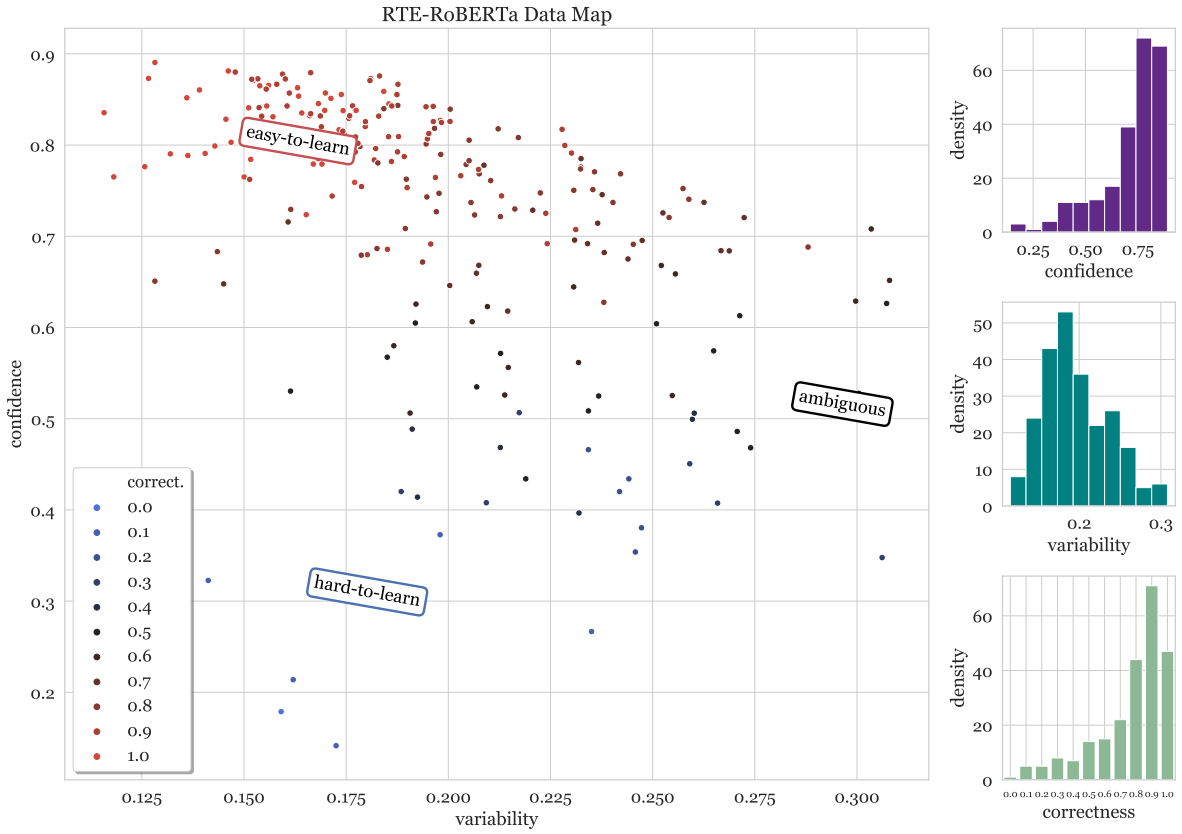

Figure 3: Data map showing location of training instances that were removed by TS-DSHAPLEY for RoBERTa on RTE.