Disentangling the Roles of Representation and Selection in Data Pruning

Yupei Du, Yingjin Song, Hugh Mee Wong, Daniil Ignatev, Albert Gatt, Dong Nguyen
Utrecht University, The Netherlands
{y.du, y.song5, h.m.wong, d.ignatev, a.gatt, d.p.nguyen}@uu.nl

Abstract

Data pruning—selecting small but impactful subsets—offers a promising way to efficiently scale NLP model training. However, existing methods often involve many different design choices, which have not been systematically studied. This limits future developments. In this work, we decompose data pruning into two key components: the *data representation* and the *selection algorithm*, and we systematically analyze their influence on the selection of instances. Our theoretical and empirical results highlight the crucial role of representations: better representations, e.g., training gradients, generally lead to a better selection of instances, regardless of the chosen selection algorithm. Furthermore, different selection algorithms excel in different settings, and none consistently outperforms the others. Moreover, the selection algorithms do not always align with their intended objectives: for example, algorithms designed for the same objective can select drastically different instances, highlighting the need for careful evaluation.

1 Introduction

A major drive of recent progress in NLP has been the scaling of training data, regarding both pretraining (Kaplan et al., 2020; Hoffmann et al., 2022; Sardana et al., 2024) and fine-tuning (Zhang et al., 2024). However, recent studies have shown that by carefully selecting a small subset of the original dataset, a process known as *data pruning*, one can train models of comparable or even better performance with much less data (Sorscher et al., 2022; Du et al., 2025; Xia et al., 2024).

Different data pruning methods exist, involving various design choices. However, no existing work has systematically studied the influence of each choice, hindering future progress. Although these methods appear diverse, we decompose them into two key components: (1) obtaining *data representations*, usually from a *reference model*, and (2) running a *selection algorithm* using these representations. Moreover, while the specific steps of selection algorithms vary, they share common *objectives*, such as maximizing the difficulty or diversity of the selected instances. Distinguishing these two components allows us to study fundamental questions: *which representations and selection objectives work better*, and *whether selection algorithms indeed meet their objectives*.

In this paper, we conduct a comprehensive study to answer these questions through both a theoretical and empirical lens. Our contributions are:

1. To study the impact of different design choices in existing data pruning methods, we conduct a comprehensive review and identify two key components: data representations and selection algorithms. Moreover, we identify three common sources for representations: *training dynamics*, e.g., loss trajectory across epochs, *hidden states*, and *gradients*; and three common selection objectives: maximizing *difficulty*, *diversity*, and *relevance* to validation data of the selected instances (§2).

2. To study which representations are more effective and why, we first identify three key criteria that effective representations should satisfy. We then theoretically analyze whether different representations meet these criteria. Finally, on both a simple synthetic task and NLP task-specific fine-tuning, we empirically validate that the representations that are more useful in theory (i.e., meet more criteria), e.g., gradients, are indeed more effective than others, e.g., hidden states (§3.1).

3. We study which selection objectives are more effective and find that no one clearly stands out: which selection objective works better depends on the context. For example, maximizing relevance to validation data excels

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 16791–16809

when a substantial train-test distribution shift is present, and maximizing difficulty works well with high data budgets. Surprisingly, *representations are more influential than selection algorithms*: when different selection algorithms use the same representation, the overlap in selected instances is greater than when the same selection algorithm is used with different representations (§3.2).

4. To gain insights into whether algorithms follow their intended objectives, we visualize their instance selection, and assess the consistency between the selections of different algorithms aiming for difficulty. Surprisingly, our results suggest that these algorithms do not always align with their objective. For instance, when maximizing difficulty, they sometimes prefer instances that are correctly predicted and far from the decision boundary; however, these are usually considered to be easier instances. Furthermore, the same objective of maximizing difficulty can lead to drastically different selections (§4).

Our findings provide actionable insights for the development of data pruning methods. Future research should: (1) develop scalable yet strong representations, and (2) carefully assess whether selection algorithms follow their intended objective.[1]

2 Representation-selection decoupling

Given a *data budget*, such as 30% of the original dataset, data pruning methods aim to select an informative subset of the data. However, it remains unclear how different design choices of these methods impact their effectiveness, because previous studies typically treat them as cohesive units. To address this gap, we identify two key components in data pruning methods: first, obtaining *representations* for each instance, such as hidden states or gradients, using a *reference model*, either off-the-shelf or fine-tuned on the original dataset; second, a *selection algorithm* to choose a subset of the data guided by a *selection objective*, such as maximizing the difficulty of the selected instances. This selected subset is then used to train the *main model*, which is the final model of interest.[2]

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

[2]We exclude methods that rely on prompting LLMs for quality scoring (Sachdeva et al., 2024; Chen et al., 2024; Lu et al., 2024; Liu et al., 2024), as these approaches add complexity through heuristic prompts and often function as black boxes, making their results difficult to interpret.

2.1 Commonly-used representations and selection objectives

Representations *Training dynamics* are widely used as a source for extracting representations, especially in fine-tuning tasks. These include metrics such as the correctness of predictions across epochs (Toneva et al., 2019), prediction probabilities of the correct class (Swayamdipta et al., 2020; Jiang et al., 2021), training error norms (Paul et al., 2021), the number of layers required for correct classification (Baldock et al., 2021), and perplexity (Moore and Lewis, 2010; Marion et al., 2023; Kwok et al., 2024). Differently, *hidden states* from pretrained language models are frequently used in pretraining scenarios (Abbas et al., 2023; Tirumala et al., 2023), because they can capture semantic information while being computationally efficient. *Gradients* are another common representation in fine-tuning. They are often used to estimate the impact of specific instances on model predictions, either through influence functions (Koh and Liang, 2017; Park et al., 2023) or training unrolling methods (Pruthi et al., 2020; Xia et al., 2024). There are also methods that use text-based features, such as bag-of-words (Canuto et al., 2018) and TF-IDF (Cunha et al., 2021).

Selection objectives After obtaining representations, various objectives are used to guide the implementation of selection algorithms. One common objective is to maximize the *difficulty* of selected instances, i.e., to select those that are harder for models to fit, as indicated by being more forgettable (Toneva et al., 2019), having a lower prediction confidence (Swayamdipta et al., 2020), a higher loss (Jiang et al., 2021; Li et al., 2024), more layers required for prediction (Baldock et al., 2021), a higher perplexity (Kwok et al., 2024), a higher self-influence (Thakkar et al., 2023), and larger distances from prototypical examples (Sorscher et al., 2022). Another objective is to maximize *diversity* in the selected data (Carbonera and Abel, 2015, 2016; Malhat et al., 2020). For example, Abbas et al. (2023) measure the similarity between instances and keep only one from each pair of highly similar instances, and Yang et al. (2024) randomly sample from different clusters of instances. Moreover, when specializing models, e.g., adapting a general model to the medical domain, it is common to maximize the *relevance* of selected instances to

validation data. For example, assuming the availability of a validation set, Xia et al. (2024) and Engstrom et al. (2024) select the most influential training instances based on training unrolling methods and influence functions, respectively.

2.2 Representative methods

Having identified commonly used representations and selection objectives, we focus on six representative methods (see Table 1, where we also include their corresponding representations and objectives). These methods cover the most common representation types: training dynamics, hidden states, and gradients, as well as key selection objectives: maximizing difficulty, diversity, and relevance to validation data. We provide an in-depth description of the methods in Appendix B.[3]

3 Deciphering the impact of different components on instance selection

This section investigates how different data representations and selection algorithms influence the selection of training instances. We first provide a theoretical analysis on how various representations differ in terms of the signals they encode (§3.1). Clarifying these differences allows us to understand the fundamental benefits and limitations of different representations, without considering specific selection algorithms. Next, we empirically compare the instances selected with different combinations of representation and selection algorithms, through experiments with both an interpretable classifier on a synthetic dataset, and the fine-tuning of language models for various tasks (§3.2).

Notation We denote the original training set with N instances as $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$. The selected subset of data is represented by $\mathcal{S} \subset \mathcal{D}$. We use B to denote the data budget (e.g., $B = |\mathcal{S}| = 0.2N$). For a data point (x_i, y_i) and a model $\mathcal{M}$, we use $p_{\mathcal{M}}(y_i|x_i)$ and $\ell_{\mathcal{M}}(x_i, y_i)$ to denote the model's prediction probability of the correct class/token and loss, $h_{\mathcal{M}}(x_i)$ to denote the last hidden state, i.e., before classifier or unembedding layer of $\mathcal{M}$. Moreover, we use $\nabla_\theta \ell_{\mathcal{M}}(x_i, y_i)$ to represent the gradient of a group of parameters θ of $\mathcal{M}$ w.r.t. $\ell_{\mathcal{M}}(x_i, y_i)$. When taking the training process of T epochs into account, we use $p^{(t)}_{\mathcal{M}}(y_i|x_i)$ and $\nabla_\theta \ell^{(t)}_{\mathcal{M}}(x_i, y_i)$ to denote $p_{\mathcal{M}}(y_i|x_i)$ and $\nabla_\theta \ell_{\mathcal{M}}(x_i, y_i)$ at epoch $t \leq T$, and η_t to denote the average learning rate of the model in epoch t.

[3] Different methods were originally proposed for specific contexts. **Our goal is not to invalidate them**, but to offer additional insights into their components.

3.1 Properties of representations

Our analysis centers on one key quantity: the *distance* between two instances, i and j, based on different representations. Measuring instance distances allows for assessing how well representations group instances of shared attributes together. Indeed, these distances are central to most methods, enabling functionalities like clustering (e.g., S2L and Prototypicality), duplicate identification (e.g., SemDeDup), and instance relevance measurement (e.g., LESS). For example, the Prototypicality method first clusters instances, and then selects instances far from the centroids as "difficult" examples. However, this raises an important question: *are the clusters good enough in separating different instances?*

Specifically, we first investigate the criteria for good representations, including the *information* representations should encode, and their *discriminative power* for different inputs, i.e., how well they separate different types of instances. We then analyze whether different representations meet these criteria. To the best of our knowledge, we are the first to systematically compare different representations in the context of data pruning.

Setup For the simplicity of analysis,[4] we consider a binary classification task with labels $y \in \{-1, +1\}$, optimized with binary cross-entropy loss, and focus on the **classification layer**. Formally, given two training instances (x_i, y_i) and (x_j, y_j), a model $\mathcal{M}$, and its classification layer w, we study the squared Euclidean distances between these two instances, computed by hidden states $h_{\mathcal{M}}(\cdot)$, losses $\ell_{\mathcal{M}}(\cdot)$, and gradients $\nabla_w \ell_{\mathcal{M}}(\cdot)$. We denote them as D_h, D_ℓ, and D_g respectively. Moreover, we use **hidden states** as the basis for our analysis, because they serve as inputs to the classification layer to compute other representations.[5]

What makes a good representation? To select a minimal subset of training data while preserving generalization, we propose that the selections need to be non-redundant and diverse. Importantly,

[4] Our analysis can be extended to multi-class classification or generation tasks by considering the prediction of a specific class or token, similar to Park et al. (2023).

[5] In other words, we treat hidden states as inputs throughout the analysis, and compute quantities such as the distances between instances and the decision boundary based on them.

Selection	Representations		
	Training dynamics	**Hidden states**	**Gradients**
Max. diversity	SmallToLarge (S2L) (Yang et al., 2024)	SemDedup (Abbas et al., 2023)	
Max. difficulty	Hard-to-learn (Swayamdipta et al., 2020; Jiang et al., 2021; İnce et al., 2023)	Prototypicality (Sorscher et al., 2022) SemDedup (Abbas et al., 2023)	Self-Influence (SI) (Feldman and Zhang, 2020; Bejan et al., 2023)
Max. relevance			LESS (Xia et al., 2024)

Table 1: Representative methods from §2.2, categorized by their representations and selection objectives.

we argue that the redundancy and diversity here should be considered with respect to model training. Specifically, selected instances must discard less relevant examples, while being diverse enough to train a robust classifier w. This entails retaining instances (1) *close to the decision boundary*[6], as those far away are either trivial (following the representer theorem (Yeh et al., 2018)), or are mislabeled or rare outliers that destabilize training (Mindermann et al., 2022): this helps select non-redundant instances; and (2) *diverse* enough, as otherwise we are likely to obtain biased models, i.e., models that make predictions using a narrow set of rules (Tirumala et al., 2023).

We therefore argue that good representations should ensure that the distances between instances D satisfy three key criteria. First, D should *account for instances' distances to the decision boundary*. Second, D should *contain instance label information*, to help selection algorithms balance samples across different labels. Third, D should *be more discriminative for important instances*, e.g., those closer to the decision boundary. This enables selection algorithms to preserve diversity among these important samples, by identifying their differences, while deprioritizing less relevant data.

Encoded information We express gradients and losses as functions of hidden states and model parameters to study information encoded by different representations, and have the following result, for which the derivation can be found in Appendix D.

Remark 3.1 (Explicit expressions)**.** *Let* $z_* = y_* w^T h_{\mathcal{M}}(x_*)$ *be the (signed and scaled) distance from* $h_{\mathcal{M}}(x_*)$ *to the decision boundary. We have* $D_\ell = (\log((1+e^{-z_i})/(1+e^{-z_j})))^2$, $D_g = \|\frac{y_i h_{\mathcal{M}}(x_i)}{1+e^{z_i}} - \frac{y_j h_{\mathcal{M}}(x_j)}{1+e^{z_j}}\|_2^2 = \frac{\|h_{\mathcal{M}}(x_i)\|_2^2}{(1+e^{z_i})^2} + \frac{\|h_{\mathcal{M}}(x_j)\|_2^2}{(1+e^{z_j})^2} - 2\frac{y_i y_j h_{\mathcal{M}}(x_i)^T h_{\mathcal{M}}(x_j)}{(1+e^{z_i})(1+e^{z_j})}$.

We make two key observations. First, compared to the distance between hidden states (D_h), the distance between losses (D_ℓ) additionally integrates the distances to the decision boundary (i.e., z_i and z_j). Second, gradients (D_g) further reflect label agreement. Specifically, D_g is small when (1) instances are easy (i.e., z_i and z_j are large), increasing denominators; and (2) hidden states are similar when their labels agree, and vice versa, increasing the third term's numerator. These observations show that losses and gradients are stronger than hidden states, for identifying instances that are similar for the training process, because they encode instances' distances to the decision boundary. Furthermore, only gradients are label-aware.

Discriminative power We examine the *discriminative power* of the distance between two instances based on different representations: the more sensitive these distances are to the changes of inputs, i.e., hidden states here, the more discriminative they are. Specifically, we analyze how this discriminative power varies with an instance's *distance to the decision boundary*. Ideally, distances should be more discriminative for instances near the decision boundary, enabling data pruning methods to capture finer distinctions, while ignoring variations among instances further away, since they are less relevant. To quantify this, we measure the **Jacobian magnitudes** of these distances w.r.t. hidden states.

Formally, let $J_{h_{\mathcal{M}}(x_i)}(D_h)$, $J_{h_{\mathcal{M}}(x_i)}(D_\ell)$, and $J_{h_{\mathcal{M}}(x_i)}(D_g)$ be the Jacobian matrices of D_h, D_ℓ, and D_g with respect to $h_{\mathcal{M}}(x_i)$. By our distance definition, for a given representation $r(\cdot)$, the distance between two instances is defined as $D_r = \|r(x_i, y_i) - r(x_j, y_j)\|_2^2$. We can then write

[6]The decision boundary of the final model, which we approximate using that of the reference model.

the Jacobian of D_r with respect to $h_{\mathcal{M}}(x_i)$ as

$$J_{h_{\mathcal{M}}(x_i)}(D_r) = \frac{\partial D_r}{\partial r(x_i, y_i)} \frac{\partial r(x_i, y_i)}{\partial h_{\mathcal{M}}(x_i)}$$
$$= 2J_{h_{\mathcal{M}}(x_i)}\big(r(x_i, y_i)\big)^\top \big(r(x_i, y_i) - r(x_j, y_j)\big).$$

Here we can see the Jacobian is influenced by the distance value through $r(x_i, y_i) - r(x_j, y_j)$. However, our goal here is to quantify how inputs' distances to the decision boundary (based on hidden states) relate to D_r's discriminative power, independent of the specific distance value. Therefore, *we focus on* $\mathbf{J}_{\mathbf{h}_{\mathcal{M}}(\mathbf{x_i})}\big(\mathbf{r}(\mathbf{x_i}, \mathbf{y_i})\big)$, and use the **spectral norm** to measure its magnitude. Formally:

Definition 3.2 (Discriminative power)**.** *We define the discriminative power of losses and gradients as the spectral norms of their Jacobian w.r.t.* $h_{\mathcal{M}}(x_i)$*:*

$$\mathcal{C}_\ell = \|J_{h_{\mathcal{M}}(x_i)}(\ell_{\mathcal{M}}(x_i, y_i))\|,$$
$$\mathcal{C}_g = \|J_{h_{\mathcal{M}}(x_i)}(\nabla_w \ell_{\mathcal{M}}(x_i, y_i))\|,$$

where $\|\cdot\|$ *denotes the spectral norm. Analogously, we get* $\mathcal{C}_h = 1$.

Based on the above definitions we have the following results. Both proofs are in Appendix D.

Theorem 3.3 (Region dependence)**.** $\mathcal{C}_\ell$ *and* $\mathcal{C}_g$ *are dependent on inputs' distances to the decision boundary, satisfying*

$$\mathcal{C}_\ell = \frac{\|w\|}{1 + e^{z_i}} = (1 - p_{\mathcal{M}}(y_i|x_i))\|w\|, \text{ and}$$
$$\mathcal{C}_g \leq \frac{1}{1 + e^{z_i}} + \frac{e^{z_i}}{(1 + e^{z_i})^2}\|h_{\mathcal{M}}(x_i)\|\|w\|.$$

Corollary 3.4. *Let* $\alpha := \|w\|\|h_{\mathcal{M}}(x_i)\|$*. When* α *is smaller than the positive root of* $-x(1 - e^x) = 1 + e^x$ *(approximately 1.544),* $\mathcal{C}_g$ *decreases monotonically as* z_i *increases, similar to* $\mathcal{C}_\ell$*. However, when* α *is larger,* $\mathcal{C}_g$ *increases with* z_i *for* $z_i \leq \log\left(\frac{\alpha-1}{\alpha+1}\right)$*, and decreases for* $z_i > \log\left(\frac{\alpha-1}{\alpha+1}\right)$.

Remark 3.5. *Theorem 3.3 shows that,* $\mathcal{C}_\ell$ *monotonically decreases with* z_i *and* $p_{\mathcal{M}}(y_i|x_i)$*, i.e., the prediction probability.*

Remark 3.6. *Corollary 3.4 indicates that, when* $\alpha >\sim 1.544$*,* $\mathcal{C}_g$ *peaks at* $z_i = \log\left(\frac{\alpha-1}{\alpha+1}\right)$*, which means the corresponding data point is close to the decision boundary but misclassified. Meanwhile, when* α *is smaller,* $\mathcal{C}_g$ *decreases with* z *(and thus prediction probability), similar to* $\mathcal{C}_\ell$*.*[7]

[7]Intuitively, α reflects the magnitudes of the model's weights and the inputs' hidden states. Across all models of our experiments, we consistently find $\alpha >\sim 1.544$.

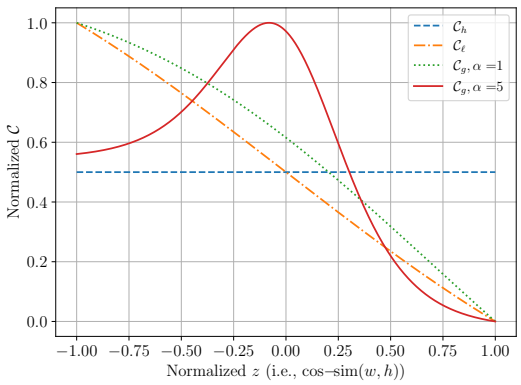


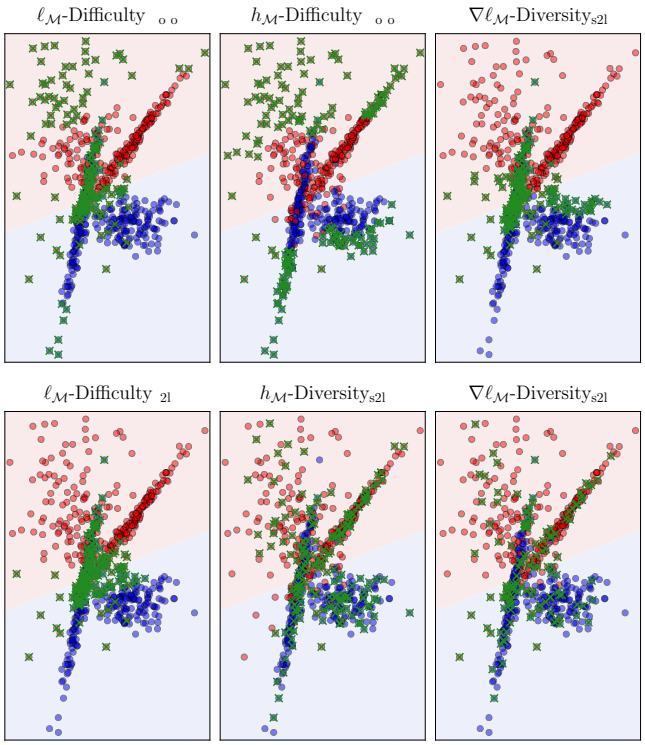
Figure 1: (Min-max normalized) discriminative power of the distance between instances, computed by different representations: the loss's discriminative power ($\mathcal{C}_\ell$) monotonically decreases, while the gradient's ($\mathcal{C}_g$) peaks near the decision boundary for large αs.

We provide a visual illustration of the discriminative power of different representations in Figure 1. In particular, we highlight the property of the gradients with $\alpha = 5$: it is discriminative when predictions are wrong, and peaks near the decision boundary, and becomes very small once the prediction is confidently correct. According to our previous analysis, *this effectively enables the selection of diverse and non-redundant examples* for learning classifiers, as they make instances near the decision boundary more distinguishable while ignoring the redundant easy ones. In contrast, algorithms that use losses will likely over-select those with a high loss (potentially destabilizing training), while distances based on hidden states are indifferent to inputs' distance to the decision boundary.

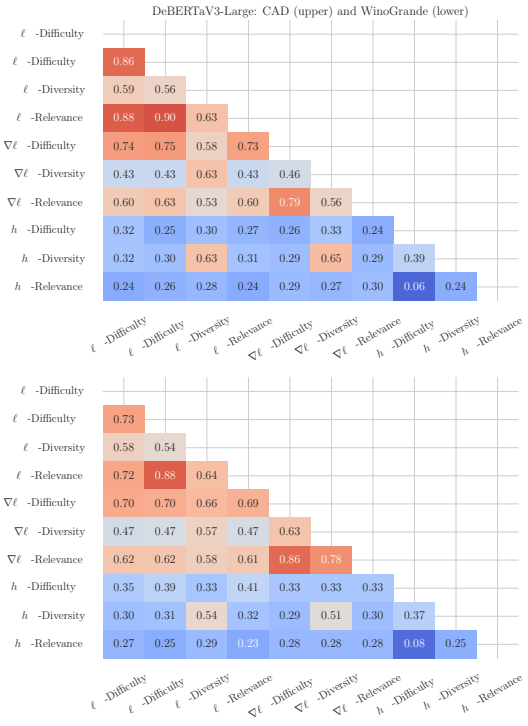
3.2 Properties of selection algorithms

Building on the insights into representations (§3.1), this section examines the properties of selection algorithms. We focus on two key aspects. First, we analyze *how changing the data representations affect the selection of instances*, to understand the joint effects of both steps and the sensitivity of selection algorithms to different representations. Second, we investigate *whether selection algorithms indeed follow their objectives*, by visualizing the selections and comparing the overlap between selection algorithms with the same objective.

Setup We focus on three selection algorithms with different objectives from §2.2: (1) prioritizing difficulty, as in prototypicality (**difficulty$_{\text{proto}}$**); (2) prioritizing diversity, as in S2L (**diversity$_{\text{s2l}}$**); and



(a) Selections on synthetic data.



(b) Selection consistency on NLP tasks.

Figure 2: The consistency of selections across different representations and selection algorithms. (a) Synthetic data: we generate 600 data points from a 2D Gaussian mixture model with red and blue data points representing two classes. The different background colors visualize the decision boundary of the logistic regression reference model. The green Xs are the selected data points (30% of the data). (b) NLP tasks: we use different methods to select 30% of the data points and compute their overlapping ratios (i.e., $|\mathcal{S}_1 \cap \mathcal{S}_2|/|\mathcal{S}_1|$ for two subsets $\mathcal{S}_1$ and $\mathcal{S}_2$). Here we show the results for DeBERTaV3-Large on CAD and WinoGrande.

(3) prioritizing relevance to validation data, as in LESS (**relevance$_{\text{less}}$**). We combine each selection algorithm with all three representations. We also compare Hard-to-Learn (**difficulty$_{\text{htl}}$**) with prototypicality, because both methods aim to select *difficult* instances. We use 30% of the data as our budget.

First, we conduct a synthetic experiment to provide an interpretable analysis. We use a 2D Gaussian mixture model to generate 600 data points, which we treat as the hidden states. We then train a logistic regression classifier as the reference model to collect training dynamics and gradients. We visualize the selected instances in Figure 2a.

Second, we conduct task-specific fine-tuning experiments on three different types of tasks: CAD (binary hate speech classification, Vidgen et al., 2021), for which we also include DynaHate as an OOD test set (Kiela et al., 2021), WinoGrande (multiple choice commonsense reasoning, Sakaguchi et al., 2021), and DialogSum (abstractive summarization, Chen et al., 2021). We use DeBERTaV3 base and large (He et al., 2023) for CAD and WinoGrande, and OPT 125M and 350M (Zhang et al., 2022) for DialogSum.[8] We show the ratios of mutually selected instances between different representation-selection combinations in Figure 2b. Note that because we select 30% of the data points, random selection would result in an overlap ratio of 0.3.[9][10]

Varying representations drastically changes selection For example, $h_{\mathcal{M}}$-difficulty$_{\text{proto}}$ and $\ell_{\mathcal{M}}$-difficulty$_{\text{proto}}$ on synthetic data respectively select instances far from and near the decision boundary (Figure 2a), and $h_{\mathcal{M}}$-relevance$_{\text{less}}$ and $\nabla_w\ell_{\mathcal{M}}$-relevance$_{\text{less}}$ have lower-than-random overlap on

[8] We use relatively small models to avoid huge computation during both training (we trained 1200+ models for controlled comparisons) and gradient projection (which can take > 10 times longer than training due to high dimensionality).

[9] Let N be the total size of the dataset. The expected number of overlap items is $|\mathcal{S}_1 \cap \mathcal{S}_2| = 0.3N \times 0.3N = 0.09N$. Since $|\mathcal{S}_1| = 0.3N$, the overlap ratio is $0.09N/0.3N = 0.3$.

[10] We focus on settings where the validation set come from the same distribution as the test set. However, we also include a discussion of the out-of-distribution settings in Appendix C.

both NLP tasks (Figure 2b). Nevertheless, we find the the sensitivity of selection algorithms towards representations varies. Particularly, *diversity-preserving algorithms are less affected by the representation choice*. For instance, compared to prototypicality using different representations, diversity shows smaller variations in Figure 2a, and similar results are observed in the NLP tasks in Figure 2b. This is consistent with that diversity-preserving algorithms sample evenly from different regions.

Representations are more influential than the selection algorithms themselves For example, $\nabla_w \ell_{\mathcal{M}}$-difficulty$_{\text{proto}}$ overlaps more on CAD with $\nabla_w \ell_{\mathcal{M}}$-diversity$_{\text{s2l}}$ (0.46) and $\nabla_w \ell_{\mathcal{M}}$-relevance$_{\text{less}}$ (0.79), than with $h_{\mathcal{M}}$-difficulty$_{\text{proto}}$ (0.26), see Figure 2b. Additionally, the selections based on gradients and losses have larger overlap with each other than with those based on hidden states. For example, the overlap when using gradients and losses with the same selection algorithm ($\nabla_w \ell_{\mathcal{M}}$-difficulty$_{\text{proto}}$ and $\ell_{\mathcal{M}}$-difficulty$_{\text{proto}}$) is as large as 0.75 on CAD. The observations here are consistent with our theoretical analysis in §3.1: losses and gradients are more informative.

Selections do not always following their objectives Because selection algorithms are typically heuristic-driven to achieve specific objectives, it is crucial to assess whether they indeed follow these objectives. Surprisingly, our results suggest otherwise: (1) On synthetic data, most selections from $h_{\mathcal{M}}$-difficulty$_{\text{proto}}$—which aims to select difficult instances—are those that are correctly predicted and far from the decision boundary, which can be considered to be the easier ones. (2) Comparing the selections under the same objective, we observe that *these selections can be vastly different*. For example, even though they both aim to select difficult instances, $h_{\mathcal{M}}$-difficulty$_{\text{proto}}$ and $\ell_{\mathcal{M}}$-difficulty$_{\text{htl}}$ show very low consistency in instance selections: this divergence is evident in synthetic data, where only a few data points are mutually selected (Figure 2a); and in NLP tasks, their overlap ratios are close to the random-guess baseline (Figure 2b). This result highlights the need for carefully assessing the consistency between selection algorithms and their intended objectives, in future studies.

4 Performance Across Data Budgets

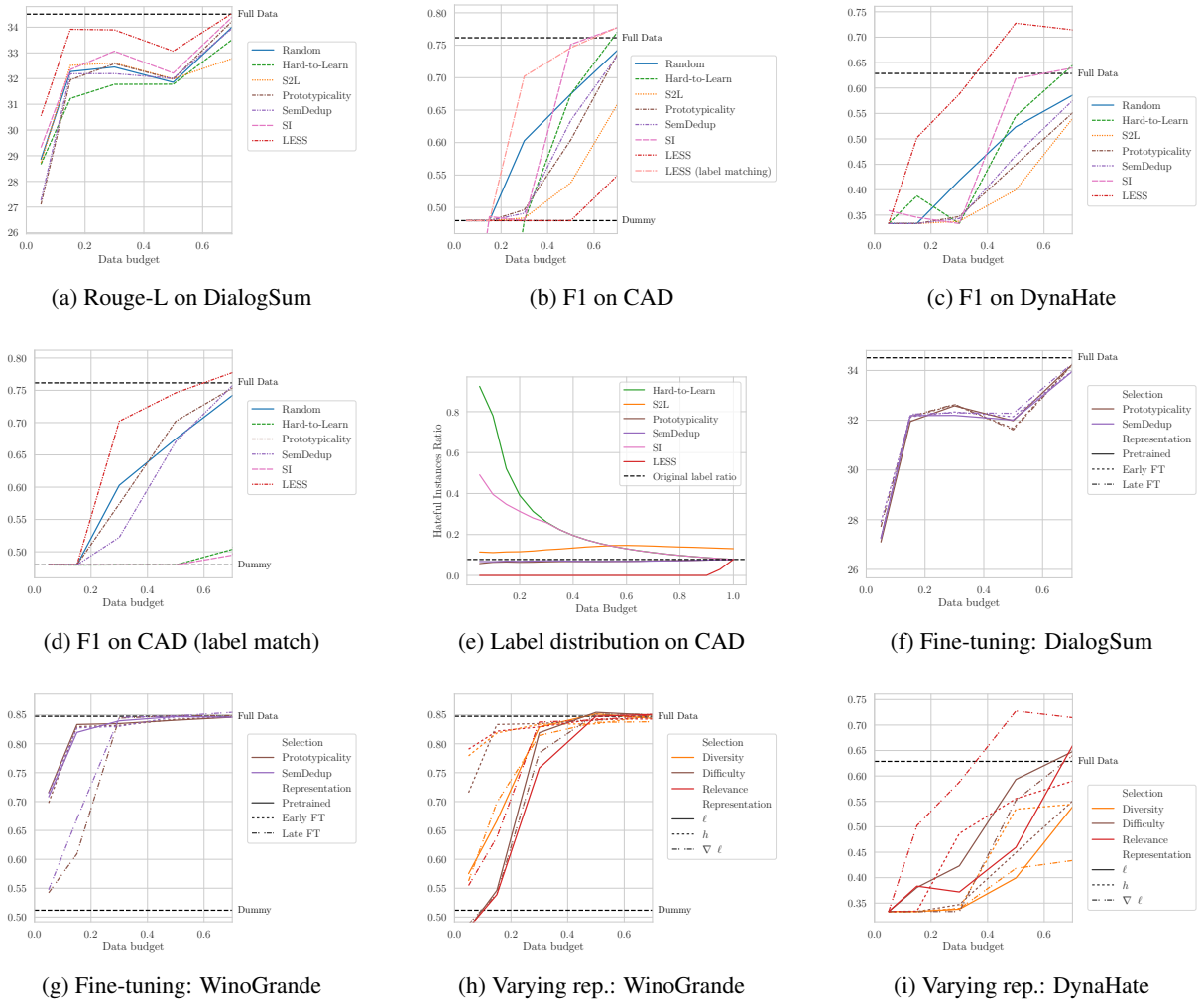
Building on our previous analyses on how different components affect data selection (§3.2), this section examines their impact on model performance under different data budgets. These experiments help validate our previous findings on the effectiveness of different representations and selection algorithms, while addressing practical questions about *which data pruning methods are best suited for specific tasks and data budgets*, such as when handling substantial distribution shift between training and testing data. Moreover, we perform two sets of ablation experiments: using fine-tuned instead of pretrained hidden states, as they may encode task-specific information; and experimenting with different representation-selection pairs, to better understand the contribution of each component.

Setup Our experiments on NLP tasks follow the same setup as in §3.2. Moreover, we use six different data budgets 5%, 15%, 30%, 50%, and 70% of the original dataset, and train all models for 15 epochs. Additionally, we consider three baselines: random selection (Rand), the full original dataset (Full Data), and a dummy predictor (Dummy), which represents the better performance between a randomized predictor and a majority class predictor. See Appendix A for more details.

4.1 Main observations

We make two main observations from our results (see Figure 3 for representative examples, with additional results in Appendix E). First, selecting the appropriate data pruning method for each specific setting is crucial: when they are applied outside their original context, *they are often outperformed by random selection*, which is consistent with Okanovic et al. (2024). Notably, hidden-state-based methods perform worse than or similarly to random selection on all tasks, especially with lower data budgets. This aligns with our previous results (§3), that pretrained hidden states may not have sufficient discriminative power to select the most important instances for the model parameters.[11] Similarly, higher data budgets are needed for other methods that aim to select difficult instances, i.e., hard-to-learn and self-influence: they achieve competitive performance with $> 30\%$ data budgets, but are only comparable to the dummy baseline with lower data budgets. This is consistent with the observations in Swayamdipta et al. (2020), that using

[11] Note that our experiments differ from previous studies that use hidden states, as they only experimented under high data budget settings (Sorscher et al., 2022) and noisy pretraining datasets (Abbas et al., 2023).



(a) Rouge-L on DialogSum (b) F1 on CAD (c) F1 on DynaHate

(d) F1 on CAD (label match) (e) Label distribution on CAD (f) Fine-tuning: DialogSum

(g) Fine-tuning: WinoGrande (h) Varying rep.: WinoGrande (i) Varying rep.: DynaHate

Figure 3: Results with DeBERTaV3$_{\text{Large}}$ and OPT-350M models. (a)–(d) show performance across data budgets, (e) presents label distributions, while (f) and (g) compare pretrained vs. fine-tuned hidden states, and (h) and (i) examine representation variation for WinoGrande and DynaHate.

only the hardest instances will make models fail to converge.

Second, gradient-based methods like LESS and self-influence perform competitively across most tasks (Figure 3a & 3c), *reaffirming the effectiveness of gradients as data representations*. Interestingly, LESS performs well on highly imbalanced datasets only when applying **label matching**, i.e., enforcing selected instances to maintain the original label ratio. Without this constraint, LESS tends to over-select majority-label training instances, because it selects training instances based on their distance to validation data. Since instances with the same label tend to have shorter distances between their gradient representations (§3.1), LESS would further amplify this bias. We validate this on CAD, where only 10% of the instances are hateful (Figure 3e), by plotting the ratio of hateful labels in selected instances. Figure 3e shows that LESS primarily selects non-hateful instances, with very few hateful ones included until the data budget reaches 90%. However, this constraint can be detrimental to methods like hard-to-learn, which prioritizes rare instances such as hateful ones.

4.2 Ablation analysis

Fine-tuned hidden states We previously observed that hidden-state-based methods perform similarly to random selection. Since effective representations should encode label information and the distance to the decision boundary, we examine whether fine-tuning can help improve their performance. Specifically, we compare two types of hidden states when models stop training at early (one epoch, retaining more pretrained knowledge) and late (15 epochs, encoding more task-specific information) stages, as shown in Figure 3f and 3g. However, there is little difference between using

different hidden states, with none of them clearly outperforming random selection, suggesting the insufficiency of fine-tuning.

Varying representations We have shown that data representations have a greater influence on selections than the selection algorithms themselves (§3). To study whether this holds for model performance, we similarly use different representations with $\text{diversity}_{\text{s2l}}$, $\text{difficulty}_{\text{proto}}$, and $\text{relevance}_{\text{less}}$, and show the results for $\text{DeBERTaV3}_{\text{Large}}$ on WinoGrande and DynaHate in Figures 3h & 3i. In line with §3.2, we find that *similar representations often lead to similar performance* (Figure 3h). Nevertheless, the best selection algorithm is still task-dependent. For example, when there is a train-test distribution mismatch (e.g., DynaHate in Figure 3i), LESS performs better than other methods by considering validation performance.

5 Conclusion

Despite the success of data pruning, the contributions of its design choices have remained unclear. This paper has identified two key components: data representations and selection algorithms, and provided a comprehensive overview of common choices (§2). Moreover, we have provided both theoretical and controlled empirical analyses on their effectiveness (§3), and their implications across different data budgets (§4). Our results highlight the critical role of data representations due to their impact on selected instances, and the importance of evaluating selection algorithms carefully, as they are not guaranteed to meet their objectives. Our findings stress the need for the development of efficient and informative data representations.

Limitations

One limitation of our work is its focus on task-specific fine-tuning, leaving other settings, like pre-training, supervised fine-tuning, and reinforcement learning from verifiable rewards, unexplored. This is largely due to (1) the large amount of computation required to conduct rigorous controlled studies such as ours, and (2) the challenges in scalable and low-cost evaluation (Zheng et al., 2023). Future studies could explore data pruning approaches in these settings by generating synthetic training and validation tasks, which allows for low-cost and controlled studies. This has been recently shown to be useful for proof-of-concept studies (Allen-Zhu and Li, 2024), whose conclusions generalize to larger scale settings well. Moreover, we focus on methods that do not require external models (e.g., prompting language models to evaluate example quality). Future work could expand our analyses to include such approaches.

Acknowledgments

We thank the anonymous reviewers for their helpful comments. We thank members of the NLP group at Utrecht University for their feedback, especially Anna Wegmann. We thank the members of the MaiNLP group at LMU Munich for their feedback, especially Barbara Plank and Philipp Mondorf. This work is supported by the ERC Starting Grant DataDivers 101162980.

References

Amro Kamal Mohamed Abbas, Kushal Tirumala, Daniel Simig, Surya Ganguli, and Ari S. Morcos. 2023. Semdedup: Data-efficient learning at web-scale through semantic deduplication. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*.

Zeyuan Allen-Zhu and Yuanzhi Li. 2024. Physics of language models: Part 3.1, knowledge storage and extraction. In *Forty-first International Conference on Machine Learning*.

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*.

Irina Bejan, Artem Sokolov, and Katja Filippova. 2023. Make every example count: On the stability and utility of self-influence for learning from noisy NLP datasets. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10107–10121, Singapore. Association for Computational Linguistics.

Rajat Bhatnagar, Ananya Ganesh, and Katharina Kann. 2022. CHIA: CHoosing instances to annotate for machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 7299–7315, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Sérgio Canuto, Daniel Xavier Sousa, Marcos André Gonçalves, and Thierson Couto Rosa. 2018. A thorough evaluation of distance-based meta-features for automated text classification. *IEEE Transactions on Knowledge and Data Engineering*, 30(12):2242–2256.

Joel Luis Carbonera and Mara Abel. 2015. A density-based approach for instance selection. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 768–774.

Joel Luis Carbonera and Mara Abel. 2016. A novel density-based approach for instance selection. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 549–556.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, and Hongxia Jin. 2024. Alpagasus: Training a better alpaca with fewer data. In *The Twelfth International Conference on Learning Representations*.

Yulong Chen, Yang Liu, Liang Chen, and Yue Zhang. 2021. DialogSum: A real-life scenario dialogue summarization dataset. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 5062–5074, Online. Association for Computational Linguistics.

Washington Cunha, Vítor Mangaravite, Christian Gomes, Sérgio Canuto, Elaine Resende, Cecilia Nascimento, Felipe Viegas, Celso França, Wellington Santos Martins, Jussara M. Almeida, Thierson Rosa, Leonardo Rocha, and Marcos André Gonçalves. 2021. On the cost-effectiveness of neural and non-neural approaches and representations for text classification: A comprehensive comparative study. *Information Processing & Management*, 58(3):102481.

Yupei Du, Albert Gatt, and Dong Nguyen. 2025. FTFT: Efficient and robust fine-tuning by transferring training dynamics. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 1294–1308, Abu Dhabi, UAE. Association for Computational Linguistics.

Logan Engstrom, Axel Feldmann, and Aleksander Madry. 2024. Dsdm: Model-aware dataset selection with datamodels. In *Forty-first International Conference on Machine Learning*.

Vitaly Feldman. 2020. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 954–959, New York, NY, USA. Association for Computing Machinery.

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.

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, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. 2022. Training compute-optimal large language models. *Preprint*, arXiv:2203.15556.

Osman Batur İnce, Tanin Zeraati, Semih Yagcioglu, Yadollah Yaghoobzadeh, Erkut Erdem, and Aykut Erdem. 2023. Harnessing dataset cartography for improved compositional generalization in transformers. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *Preprint*, arXiv:2311.10702.

Ziheng Jiang, Chiyuan Zhang, Kunal Talwar, and Michael C Mozer. 2021. Characterizing structural regularities of labeled data in overparameterized models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5034–5044. PMLR.

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*.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, Zhiyi Ma, Tristan Thrush, Sebastian Riedel, Zeerak Waseem, Pontus Stenetorp, Robin Jia, Mohit Bansal, Christopher Potts, and Adina Williams. 2021. Dynabench: Rethinking benchmarking in NLP. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124, Online. Association for Computational Linguistics.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Devin Kwok, Nikhil Anand, Jonathan Frankle, Gintare Karolina Dziugaite, and David Rolnick. 2024. Dataset difficulty and the role of inductive bias. *Preprint*, arXiv:2401.01867.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. 2024. From quantity to quality: Boosting LLM performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human*

Language Technologies (Volume 1: Long Papers), pages 7595–7628, Mexico City, Mexico. Association for Computational Linguistics.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. 2024. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *The Twelfth International Conference on Learning Representations*.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. 2024. #instag: Instruction tagging for analyzing supervised fine-tuning of large language models. In *The Twelfth International Conference on Learning Representations*.

Mohamed Malhat, Mohamed El Menshawy, Hamdy Mousa, and Ashraf El Sisi. 2020. A new approach for instance selection: Algorithms, evaluation, and comparisons. *Expert Systems with Applications*, 149:113297.

Max Marion, Ahmet Üstün, Luiza Pozzobon, Alex Wang, Marzieh Fadaee, and Sara Hooker. 2023. When less is more: Investigating data pruning for pretraining llms at scale. *Preprint*, arXiv:2309.04564.

Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, and Yarin Gal. 2022. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 15630–15649. PMLR.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 220–224, Uppsala, Sweden. Association for Computational Linguistics.

Patrik Okanovic, Roger Waleffe, Vasilis Mageirakos, Konstantinos Nikolakakis, Amin Karbasi, Dionysios Kalogerias, Nezihe Merve Gürel, and Theodoros Rekatsinas. 2024. Repeated random sampling for minimizing the time-to-accuracy of learning. In *The Twelfth International Conference on Learning Representations*.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. 2023. TRAK: Attributing model behavior at scale. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 27074–27113. PMLR.

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*.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, volume 33, pages 19920–19930. Curran Associates, Inc.

Noveen Sachdeva, Benjamin Coleman, Wang-Cheng Kang, Jianmo Ni, Lichan Hong, Ed H. Chi, James Caverlee, Julian McAuley, and Derek Zhiyuan Cheng. 2024. How to train data-efficient llms. *Preprint*, arXiv:2402.09668.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: an adversarial winograd schema challenge at scale. *Commun. ACM*, 64(9):99–106.

Nikhil Sardana, Jacob Portes, Sasha Doubov, and Jonathan Frankle. 2024. Beyond chinchilla-optimal: Accounting for inference in language model scaling laws. In *Forty-first International Conference on Machine Learning*.

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*.

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.

Megh Thakkar, Tolga Bolukbasi, Sriram Ganapathy, Shikhar Vashishth, Sarath Chandar, and Partha Talukdar. 2023. Self-influence guided data reweighting for language model pre-training. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2033–2045, Singapore. Association for Computational Linguistics.

Kushal Tirumala, Daniel Simig, Armen Aghajanyan, and Ari Morcos. 2023. D4: Improving llm pretraining via document de-duplication and diversification. In *Advances in Neural Information Processing Systems*, volume 36, pages 53983–53995. Curran Associates, Inc.

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*.

Bertie Vidgen, Dong Nguyen, Helen Margetts, Patricia Rossini, and Rebekah Tromble. 2021. 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.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. 2024. LESS: Selecting influential data for targeted instruction tuning. In *ICLR 2024 Workshop on Navigating and Addressing Data Problems for Foundation Models*.

Yu Yang, Siddhartha Mishra, Jeffrey N Chiang, and Baharan Mirzasoleiman. 2024. Smalltolarge (s2l): Scalable data selection for fine-tuning large language models by summarizing training trajectories of small models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. 2018. Representer point selection for explaining deep neural networks. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Biao Zhang, Zhongtao Liu, Colin Cherry, and Orhan Firat. 2024. When scaling meets LLM finetuning: The effect of data, model and finetuning method. In *The Twelfth International Conference on Learning Representations*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A Experimental Details

Implementation Details All experiments were conducted using the AdamW optimizer. For most models, we set the learning rate to 2e-5, except for DeBERTaV3$_{\text{Large}}$, where we followed He et al. (2023) and used 1e-5. Additionally, we do a learning rate warmup for the first 10% of training steps. For gradient-based pruning, the reference models were trained with LoRA, using a higher learning rate of 1e-4, $r = 64$, and $\alpha = 16$ following Ivison et al. (2023), and apply LoRA on all linear layers. We train all models for 15 epochs, For batch size, we used 16 for both WinoGrande and DialogSum, and 32 for CAD, to fit all experiments on a single NVIDIA A100-40GB GPU. We use maximum sequence lengths of 300, 128, and 512 tokens. For all experiments we use the same reference models as the main models for fair comparison.

For k-Means clustering in S2L, Prototypicality, and SemDedup, we use 100 clusters on CAD and DialogSum, and 200 clusters on WinoGrande, following the suggestions from Tirumala et al. (2023) to set the number of clusters to around the square root of the number of instances. Moreover, we compute gradients using the first five checkpoints for all experiments, and project them into a 1024-dimensional space using Park et al. (2023) (details see hyperparameter search).

Evaluation Metrics We evaluated CAD and DynaHate using the macro F1 score, WinoGrande by accuracy, and DialogSum by ROUGE-1, ROUGE-2, and ROUGE-L (from HuggingFace Evaluate), following the original studies (Vidgen et al., 2021; Sakaguchi et al., 2021; Chen et al., 2021).

Infrastructure All experiments were run on a single NVIDIA A100-40GB GPU using three random seeds. We used PyTorch 2.3, Transformers 4.42, and vLLM 0.5 for training and inference. Moreover, we use `bfloat16` on all experiments to improve efficiency.

Hyperparameter Search We searched for four hyperparameters: the number of training epochs, the number of clusters for k-Means clustering, the dimensionality of the projected gradients, and the checkpoints to use for gradient computation.

For the number of training epochs, we first perform a search over 3, 5, 7, and 10 epochs on all datasets and models, using three random seeds. We observe that models of different sizes share similar performance trends over epochs, with improvements continuing as the number of epochs increased. We therefore use the smaller models, i.e., DeBERTaV3$_{\text{Base}}$ and OPT-125M, and extend this search over 15, 20, and 25 epochs. Across all datasets, the best performance is achieved with 15 epochs.

For the number of clusters, we search over 2, 5, 10, 20, 50, 100, and 200 clusters for each dataset and model, using three random seeds. The results are highly consistent across cluster numbers. Following Tirumala et al. (2023), we use the square root of the dataset size as a guideline, settling on 100 clusters for CAD and DialogSum, and 200 for WinoGrande.

For gradients, we use smaller models (DeBERTaV3$_{\text{Base}}$ and OPT-125M) for hyperparameter search, and only one random seed (0) to avoid the high costs of computing and projecting gradients. We compute the gradients for all 15 checkpoints, and project them into 1024, 2048, and 4096 dimensions. First, we observe that different projections yield similar results, and thus choose 1024 for further experiments for efficiency. Second, we experimented with different strategies for selecting checkpoints, including the first three, the last three, the first five, the last five, and evenly spaced three and five checkpoints. Using the first checkpoints is the most consistent with using all checkpoints, with the first five yielding a minimum Spearman's rank correlation of 0.96. We therefore use the first five checkpoints for all experiments.

B Overview of Data Pruning Methods

Hard-to-Learn (training dynamics) The Hard-to-Learn method is based on a simple intuition: training instances that are **difficult** for models to fit often contain fewer regular patterns and can thus improve model generalization (Swayamdipta et al., 2020; Jiang et al., 2021). In classification tasks, the score of an instance (x_i, y_i) is defined as *the average prediction probability of the correct label across different epochs*, i.e., $\frac{1}{T}\sum_{t=1}^{T} p_{\mathcal{M}}^{(t)}(y_i|x_i)$. The main model is then trained on instances with the lowest scores. Originally proposed for classification tasks, Bhatnagar et al. (2022) and İnce et al. (2023) extend this concept to generation tasks, by replacing the minus average prediction probability with the inverse perplexity.

SmallToLarge (training dynamics) SmallToLarge (S2L; Yang et al., 2024) is proposed to select

diverse instances, to preserve full-dataset knowledge during supervised fine-tuning. Noting that similar loss trajectories indicate similar knowledge, S2L performs three steps to ensure the diversity of the selected data. First, each training instance is represented by its *cross entropy loss trajectory* observed during reference model training. S2L then performs k-means clustering on these trajectories. Finally, S2L iteratively samples from each cluster, while balancing the number of instances across clusters.

Prototypicality (hidden states) The Prototypicality method (Sorscher et al., 2022) selects **difficult** instances in pretraining, by exploiting their *similarities*: it measures difficulty based on how *prototypical* an instance is. Specifically, after representing instances by their **hidden states**, prototypicality applies k-means clustering and ranks instances based on their distances to their cluster centroids. Instances with larger distances are considered less prototypical and therefore more difficult, and are thus selected to train the main model.

SemDeDup (hidden states) Building on Prototypicality, targeting large-scale pretraining, SemDeDup includes an additional step to also account for data diversity (Abbas et al., 2023): after clustering, it identifies semantically duplicate pairs of instances within each cluster using cosine similarities of their **hidden states**. For each identified duplicate pair, it retains the instance that lies farther from the cluster centroid, thereby prioritizing **diversity** while maintaining **difficulty**.

LESS (gradients) Proposed for supervised fine-tuning, LESS additionally requires a validation set to select more **relevant** instances, using cosine similarities between **gradients** (Xia et al., 2024). Formally, the relevance of (x_i, y_i) w.r.t. a validation instance $(x_{\text{val}}, y_{\text{val}})$ is defined as $\sum_{t=1}^{T} \eta_t \cdot \nabla_\theta \ell_{\mathcal{M}}^{(t)}(x_i, y_i)^\top \nabla_\theta \ell_{\mathcal{M}}^{(t)}(x_{\text{val}}, y_{\text{val}}))$, where η_t is the average learning rate between the t-th and the $t+1$-th checkpoint. These gradients are normalized in generation tasks because their norms negatively correlate with sequence lengths.

Self-Influence (gradients) Feldman and Zhang (2020) define memorization during training as the prediction probability decrease of an instance before and after removing it from the training set, i.e., self-influence. They argue that memorized instances are usually **difficult**-to-predict, and thus contribute more to generalization under the long-tail assumption of testing cases (Feldman, 2020). In this work, following Bejan et al. (2023), we use TracIn (Pruthi et al., 2020) for approximation. Formally, the self-influence score of (x_i, y_i) is estimated as $\sum_{t=1}^{T} \eta_t \nabla_\theta \ell_{\mathcal{M}}^{(t)}(x_i, y_i)^\top \nabla_\theta \ell_{\mathcal{M}}^{(t)}(x_i, y_i)$.

C Discussion of out-of-distribution settings

We focus on in-distribution settings in this paper, where the validation and test data come from the same distribution, although the train data may come from a different distribution, e.g., CAD(train)-DynaHate(validation/test) in HSD. Here we discuss the potential impact of the out-of-distribution (OOD) settings, where the training and validation data come from different distributions.

Regarding representations, we expect they perform similarly as in in-distribution settings. For example, gradients should be better: their discriminative power can help model build robust decision boundaries, in both in-distribution and out-of-distribution settings; meanwhile, we still expect hidden states to be less effective, because they lack the discriminative power to distinguish instances that offer different signals for model training.

Regarding selection objectives, we expect methods that prioritize difficult instances to perform better, because easy training instances usually contain more regularities and shortcuts. For example, in NLI, contradiction with negation words usually are considered "easier" than the ones without negation words, because they are seen more frequently in the training data; meanwhile, we expect methods that prioritize relevance to perform worse, because this might drive the training data distribution further away from the test data distribution, since the validation sets that guide these selections follow different distributions than the test sets.

D Proofs

D.1 Derivation of Remark 3.1

We first restate the remark for reference.

Remark D.1 (Explicit expressions)**.** *Let* $z_* = y_* w^T h_{\mathcal{M}}(x_*)$ *be the (signed and scaled) distance from* $h_{\mathcal{M}}(x_*)$ *to the decision boundary. We have* $D_\ell = (\log((1+e^{-z_i})/(1+e^{-z_j})))^2$, $D_g = \|\frac{y_i h_{\mathcal{M}}(x_i)}{1+e^{z_i}} - \frac{y_j h_{\mathcal{M}}(x_j)}{1+e^{z_j}}\|_2^2 = \frac{\|h_{\mathcal{M}}(x_i)\|_2^2}{(1+e^{z_i})^2} + \frac{\|h_{\mathcal{M}}(x_j)\|_2^2}{(1+e^{z_j})^2} - 2\frac{y_i y_j h_{\mathcal{M}}(x_i)^T h_{\mathcal{M}}(x_j)}{(1+e^{z_i})(1+e^{z_j})}$.

Derivation. We first derive the expression for D_ℓ.

$$
\begin{aligned}
D_\ell &:= \|\ell_{\mathcal{M}}(x_i, y_i) - \ell_{\mathcal{M}}(x_j, y_j)\|_2^2 && (1)\\
&= \|\log(1+e^{-y_i w^T h_{\mathcal{M}}(x_i)}) \\
&\qquad - \log(1+e^{-y_j w^T h_{\mathcal{M}}(x_j)})\|_2^2 && (2)\\
&= (\log(1+e^{-z_i}) - \log(1+e^{-z_j}))^2 && (3)\\
&= (\log(\frac{1+e^{-z_i}}{1+e^{-z_j}}))^2 && (4)
\end{aligned}
$$

Next, we derive the expression for D_g. Recall that in §3, we defined $D_g := \|\nabla_w \ell_{\mathcal{M}}(x_i, y_i) - \nabla_w \ell_{\mathcal{M}}(x_j, y_j)\|_2^2$. We first derive $\nabla_w \ell_{\mathcal{M}}(x_i, y_i) = \frac{y_i h_{\mathcal{M}}(x_i)}{1+e^{z_i}}$.

$$
\begin{aligned}
\nabla_w \ell_{\mathcal{M}}(x_i, y_i) &= \nabla_w \log(1+e^{-y_i w^T h_{\mathcal{M}}(x_i)}) && (5)\\
&= \nabla_w \log(1+e^{-z_i}) && (6)\\
&= -(1-\sigma(z_i)) y_i h_{\mathcal{M}}(x_i) && (7)\\
&= -\frac{y_i h_{\mathcal{M}}(x_i)}{1+e^{z_i}} && (8)
\end{aligned}
$$

The derivation of $\nabla_w \ell_{\mathcal{M}}(x_j, y_j) = \frac{y_j h_{\mathcal{M}}(x_j)}{1+e^{z_j}}$ is analogous to the one above, and thus we get $D_g = \|\frac{y_i h_{\mathcal{M}}(x_i)}{1+e^{z_i}} - \frac{y_j h_{\mathcal{M}}(x_j)}{1+e^{z_j}}\|_2^2$. □

D.2 Proof of Theorem 3.3

We first restate the theorem for reference.

Theorem D.2. *The discriminative power of losses and gradients (relative to that of hidden states) are dependent of the region the hidden states lie in, satisfying*

$$
\begin{aligned}
&\mathcal{C}_\ell = \frac{e^{z_i}}{1+e^{z_i}}\|w\|, \text{ and}\\
&\mathcal{C}_g \le \frac{1}{1+e^{z_i}} + \frac{e^{z_i}}{(1+e^{z_i})^2}\|h_{\mathcal{M}}(x_i)\|\|\|w\|.
\end{aligned}
$$

Proof. We first prove the discriminative power of losses. Let $\sigma(z_i) = \frac{1}{1+e^{-z_i}}$, we have

$$
\begin{aligned}
\frac{\partial \ell_{\mathcal{M}}(x_i, y_i)}{\partial h_{\mathcal{M}}(x_i)} &= -\frac{\partial \log \sigma(z_i)}{\partial z_i}\frac{\partial z_i}{\partial h_{\mathcal{M}}(x_i)} && (9)\\
&= -(1-\sigma(z_i)) y_i w && (10)\\
&= -\frac{1}{1+e^{z_i}} w && (11)\\
&= -(1-p_{\mathcal{M}}(y_i|x_i)) w. && (12)
\end{aligned}
$$

Therefore,

$$
\mathcal{C}_\ell = \|\frac{\partial \ell_{\mathcal{M}}(x_i, y_i)}{\partial h_{\mathcal{M}}(x_i)}\| = \frac{1}{1+e^{z_i}}\|w\|. \quad (13)
$$

Next, we prove the discriminative power of gradients. We have

$$
\begin{aligned}
&\frac{\partial \nabla_w \ell_{\mathcal{M}}(x_i, y_i)}{\partial h_{\mathcal{M}}(x_i)} && (14)\\
=&\frac{\partial((y_i h_{\mathcal{M}}(x_i))/(1+e^{z_i}))}{\partial h_{\mathcal{M}}(x_i)} && (15)\\
=&\frac{y_i}{1+e^{z_i}} I - \frac{y_i h_{\mathcal{M}}(x_i) e^{z_i}}{(1+e^{z_i})^2}\frac{\partial z}{\partial h_{\mathcal{M}}(x_i)} && (16)\\
=&\frac{y_i}{1+e^{z_i}} I - \frac{e^{z_i}}{(1+e^{z_i})^2} h_{\mathcal{M}}(x_i) w^\top. && (17)
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
\mathcal{C}_g &= \|\frac{\partial \nabla_w \ell_{\mathcal{M}}(x_i, y_i)}{\partial h_{\mathcal{M}}(x_i)}\| && (18)\\
&= \|\frac{1}{1+e^{z_i}} I - \frac{e^{z_i}}{(1+e^{z_i})^2} h_{\mathcal{M}}(x_i) w^\top\| && (19)\\
&\le \|\frac{1}{1+e^{z_i}} I\| + \|\frac{e^{z_i}}{(1+e^{z_i})^2} h_{\mathcal{M}}(x_i) w^\top\| && (20)\\
&= \frac{1}{1+e^{z_i}} + \frac{e^{z_i}}{(1+e^{z_i})^2}\|h_{\mathcal{M}}(x_i)\|\|w\|. && (21)
\end{aligned}
$$

□

D.3 Proof of Corollary 3.4

We first restate the corollary for reference.

Corollary D.3. *Let* $\|w\|\|h_{\mathcal{M}}(x_i)\| = \alpha$*. When* α *is smaller than the positive root of function* $-x(1-e^x) = 1+e^x$ *(*~ 1.544*),* $\mathcal{C}_g$ *monotonically decreases as* z_i *increases, which is similar to* $\mathcal{C}_\ell$*. However, when* α *is larger,* $\mathcal{C}_g$ *increases with* z_i *when* $z_i \le \log(\frac{\alpha-1}{\alpha+1})$*, which is negatively close to the decision boundary, and decreases when* $z_i > \log(\frac{\alpha-1}{\alpha+1})$.

Proof. To study the behavior of $\mathcal{C}_g$ with z_i, we take

the derivative of $\mathcal{C}_g$ with respect to z_i,

$$\frac{\partial \mathcal{C}_g}{\partial z_i} = \frac{\partial}{\partial z_i}(\frac{1}{1+e^{z_i}} + \frac{e^{z_i}}{(1+e^{z_i})^2}\alpha) \tag{22}$$

$$= -\frac{e^{z_i}}{(1+e^{z_i})^2} + \frac{e^{z_i}}{(1+e^{z_i})^2}\alpha - \frac{2e^{2z_i}}{(1+e^{z_i})^3}\alpha$$

$$= \frac{e^{z_i}}{(1+e^{z_i})^2}(-1+\alpha - \frac{2e^{z_i}}{1+e^{z_i}}\alpha)$$

$$= \frac{e^{z_i}}{(1+e^{z_i})^3}(\alpha(1-e^{z_i}) - (1+e^{z_i}))$$

$$= \frac{e^{z_i}}{(1+e^{z_i})^3}((\alpha-1) - (\alpha+1)e^{z_i}). \tag{23}$$

For the domain of z_i, we know that

$$z_i = y_i w^\top h_{\mathcal{M}}(x_i) = \alpha\cos(\phi) \in [-\alpha, \alpha] \tag{24}$$

, where ϕ is the angle between w and $h_{\mathcal{M}}(x_i)$.

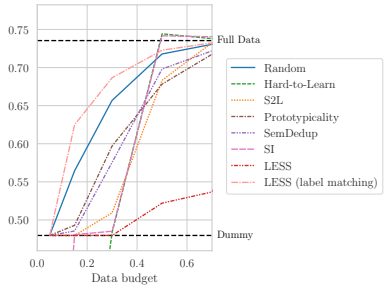
When $z_i > 0$, i.e., the prediction is correct, $\partial \mathcal{C}_g / \partial z_i < 0$. Therefore, $\mathcal{C}_g$ monotonically decreases as z_i increases.

When $z_i < 0$, i.e., the prediction is incorrect:

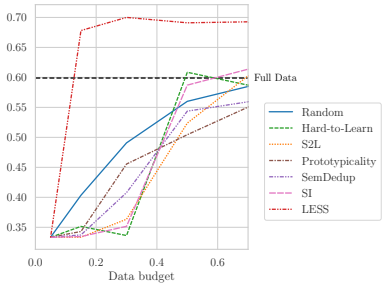
- If $0 < \alpha \leq 1$, $\alpha - 1 \leq 0$. From Eq. 23, we know that $\partial \mathcal{C}_g / \partial z_i < 0$. Therefore, $\mathcal{C}_g$ monotonically decreases as z_i increases.
- If $\alpha > 1$, when $z_i \leq \log(\frac{\alpha-1}{\alpha+1})$, $\partial \mathcal{C}_g / \partial z_i > 0$. Therefore, $\mathcal{C}_g$ increases with $z_i \in [-\alpha, \log(\frac{\alpha-1}{\alpha+1}))$ (if exists), then decreases when $z_i \geq \log(\frac{\alpha-1}{\alpha+1})$. To make sure the range exists, we need $\log(\frac{\alpha-1}{\alpha+1}) > -\alpha$, which is equivalent to $\alpha >\sim 1.544$. Otherwise, similar to the case of $\alpha \leq 1$, $\mathcal{C}_g$ monotonically decreases as z_i increases.

□

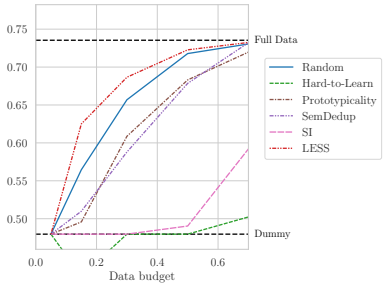
E Additional Results



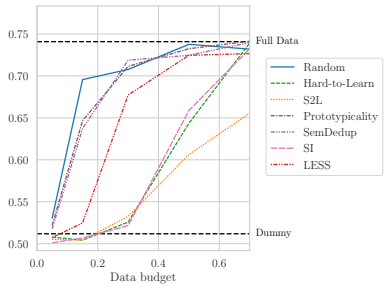
(a) F1 scores of DeBERTaV3$_{\text{Base}}$ on CAD



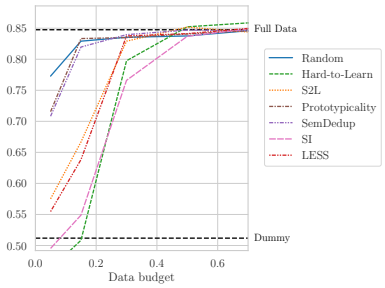
(b) F1 scores of DeBERTaV3$_{\text{Base}}$ on DynaHate



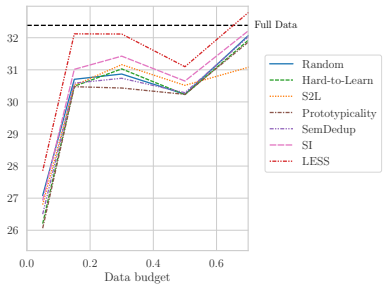
(c) F1 scores of DeBERTaV3$_{\text{Base}}$ on CAD with label balancing



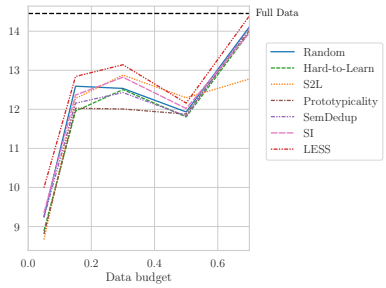
(d) Accuracy of DeBERTaV3$_{\text{Base}}$ on WinoGrande



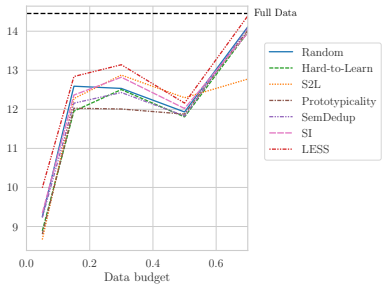
(e) Accuracy of DeBERTaV3$_{\text{Large}}$ on WinoGrande



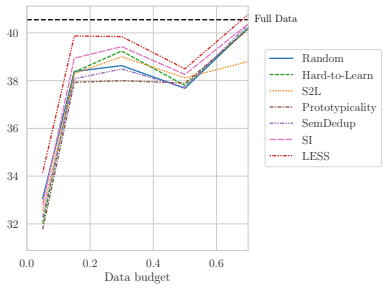
(f) Rouge-L scores of OPT-125M on DialogSum



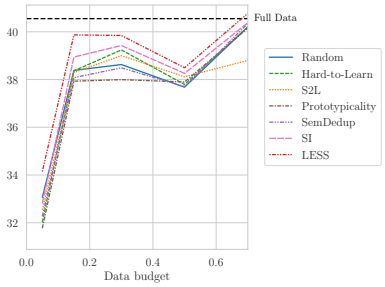
(g) Rouge-2 scores of OPT-125M on DialogSum



(h) Rouge-2 scores of OPT-350M on DialogSum

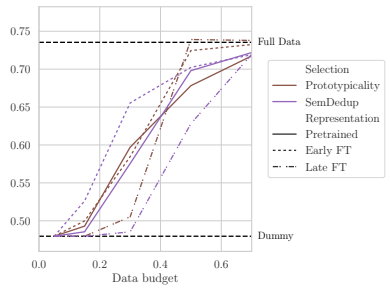


(i) Rouge-1 scores of OPT-125M on DialogSum

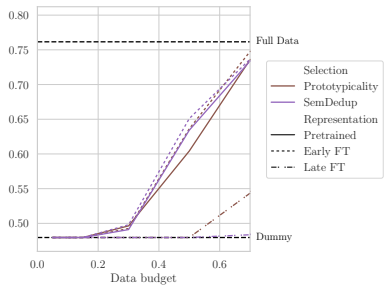


(j) Rouge-1 scores of OPT-350M on DialogSum

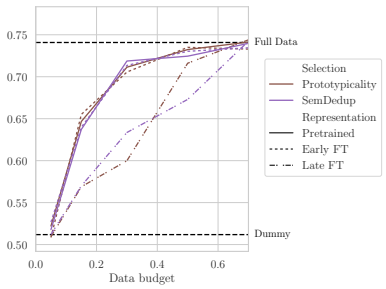
Figure 4: Model performance under different data budgets.



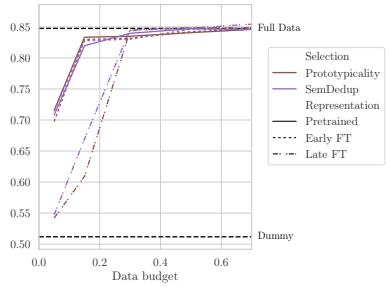
(a) Pretrained vs. Fine-Tuned (FT) Hidden States: $\text{DeBERTaV3}_{\text{Base}}$ on CAD



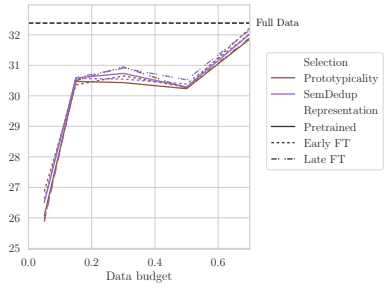
(b) Pretrained vs. Fine-Tuned (FT) Hidden States: $\text{DeBERTaV3}_{\text{Large}}$ on CAD



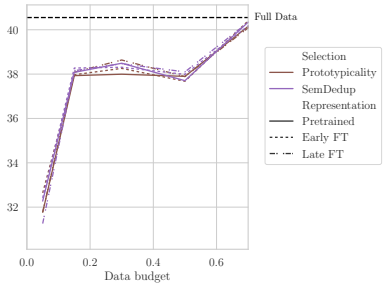
(c) Pretrained vs. Fine-Tuned (FT) Hidden States: $\text{DeBERTaV3}_{\text{Base}}$ on WinoGrande



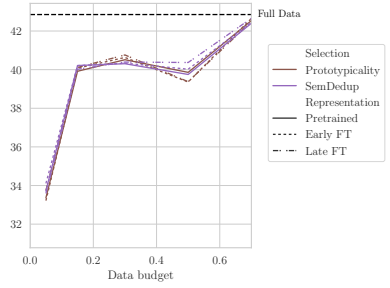
(d) Pretrained vs. Fine-Tuned (FT) Hidden States: $\text{DeBERTaV3}_{\text{Large}}$ on WinoGrande



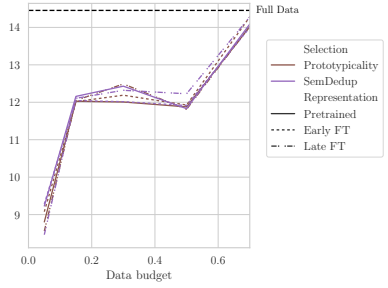
(e) Pretrained vs. Fine-Tuned (FT) Hidden States: OPT-125M on DialogSum (Rouge-L)



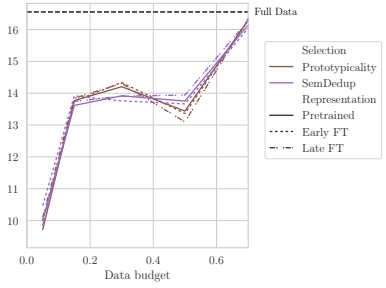
(f) Pretrained vs. Fine-Tuned (FT) Hidden States: OPT-125M on DialogSum (Rouge-1)



(g) Pretrained vs. Fine-Tuned (FT) Hidden States: OPT-350M on DialogSum (Rouge-1)

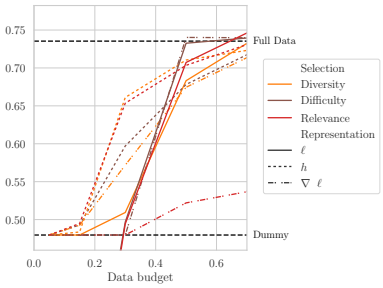


(h) Pretrained vs. Fine-Tuned (FT) Hidden States: OPT-125M on DialogSum (Rouge-2)

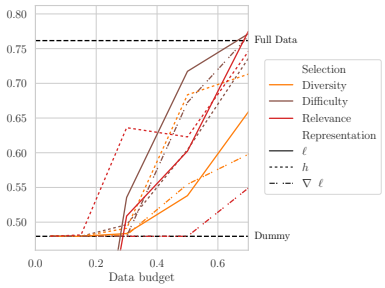


(i) Pretrained vs. Fine-Tuned (FT) Hidden States: OPT-350M on DialogSum (Rouge-2)

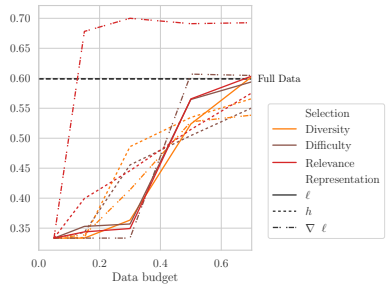
Figure 5: Ablation studies on pretrained vs. fine-tuned hidden states.



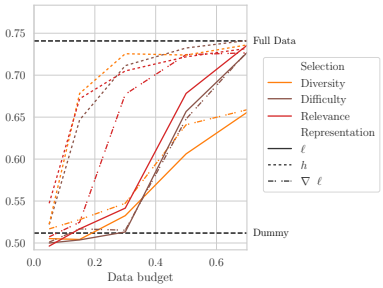
(a) DeBERTaV3$_{\text{Base}}$ on CAD (F1)



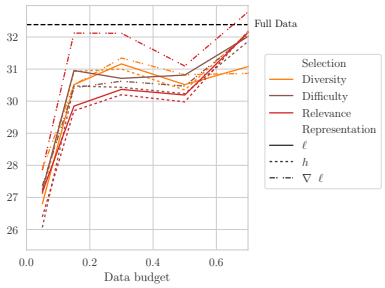
(b) DeBERTaV3$_{\text{Large}}$ on CAD (F1)



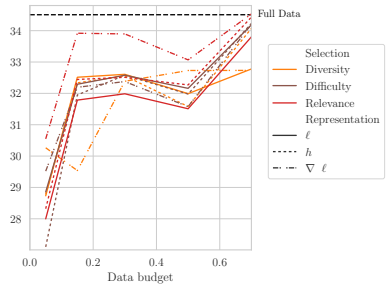
(c) DeBERTaV3$_{\text{Base}}$ on DynaHate (F1)



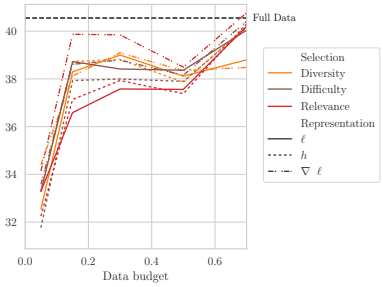
(d) DeBERTaV3$_{\text{Base}}$ on WinoGrande (Accuracy)



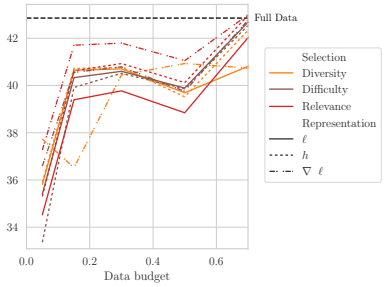
(e) OPT-125M on DialogSum (Rouge-L)



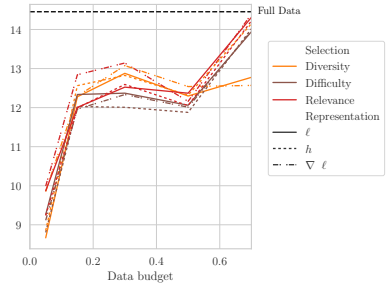
(f) OPT-350M on DialogSum (Rouge-L)



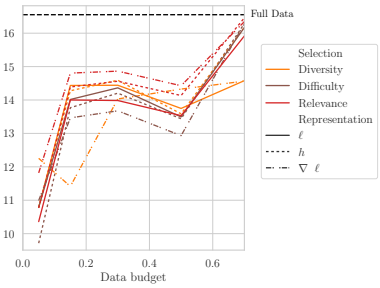
(g) OPT-125M on DialogSum (Rouge-1)



(h) OPT-350M on DialogSum (Rouge-1)



(i) OPT-125M on DialogSum (Rouge-2)



(j) OPT-350M on DialogSum (Rouge-2)

Figure 6: Ablation studies on using different data representations with the same selection algorithm.