Identifying Noise in Human-Created Datasets using Training Dynamics from Generative Models

Maeda F. Hanafi[1*], **Ishan Jindal**[2†], **Yannis Katsis**[1], **Lucian Popa**[1], **Huaiyu Zhu**[1]
[1]IBM Research, [2]Samsung R&D Institute India-Delhi,

Abstract

Instruction fine-tuning enhances the alignment of autoregressive language models (ArLMs) with human intent but relies on large-scale annotated datasets prone to label and text noise. In this paper, we show that existing noise detection techniques designed for autoencoder models (AeLMs) do not directly generalize to ArLMs due to differences in learning dynamics. We propose TDRANKER, a novel approach leveraging training dynamics to rank datapoints from easy-to-learn to hard-to-learn, effectively identifying noisy instances. Our method demonstrates robustness across multiple model architectures covering both autoencoder and autoregressive language models (GPT-2, BERT, LaMini-Cerebras-256M) and across various dataset noise levels, achieving at least 2x faster denoising than previous techniques. Applied to real-world classification and generative tasks, TDRANKER significantly improves data quality and model performance. These findings suggest that TDRANKER provides a scalable solution for refining instruction-tuning datasets, enhancing the reliability of fine-tuned ArLMs in practical applications.

1 Introduction

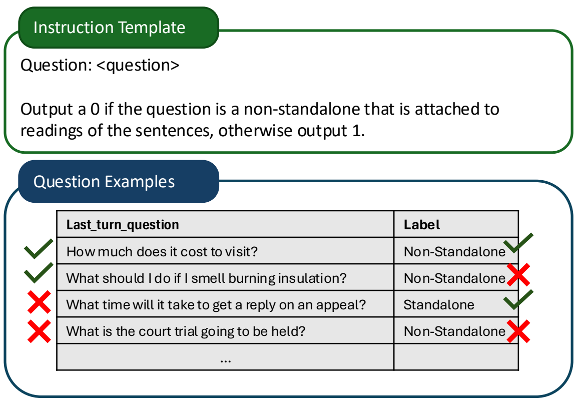
Autoregressive language models (ArLMs), also known as generative models, have recently achieved significant progress across various natural language tasks, including understanding, mathematical reasoning, and coding (Achiam et al., 2023; Team et al., 2024; Touvron et al., 2023; Roziere et al., 2023; Yang et al., 2024). These ArLMs, trained with a causal language modeling objective to predict the next token in a sequence, excel at generating coherent text but often lack alignment with human preferences (Ouyang et al., 2022). To address this, 'Instruction fine-tuning' adapts base models to better meet user needs, creating 'Instruction models' (Rafailov et al., 2024; Ethayarajh et al., 2024).

However, instruction fine-tuning is resource-intensive, requiring substantial labeled data—e.g., 10M annotated examples for LLaMa 3 instruct model (AI@Meta, 2024). This makes data collection, annotation, and training both costly and time-consuming. Moreover, the complexity of the annotation process often introduces various types of noise into the labeled datasets. Such noise can originate from human errors during labeling (*label noise*) or inherent issues in the datapoints themselves, such as grammatical mistakes or poorly constructed sentences that can even lead to a change in the intended meaning of the sentence (*text noise*). Figure 1 illustrates the various types of noise commonly found in instruction tuning datasets, including errors introduced by annotators, inconsistencies in labeling, and issues within the datapoints.

While deep learning models are generally considered robust to a certain degree of label noise (Oyen et al., 2022), studies on autoencoder language models (AeLMs), which include discriminative models like BERT, have revealed that these models face significant challenges when fine-tuned on datasets with noisy labels (Arpit et al., 2017; Zhang et al., 2021) and are susceptible to overfitting to label noise present in the dataset (Zhu et al., 2022) leading to significant performance loss on downstream tasks. This made it essential to either (1) develop strategies for fine-tuning AeLMs with such noisy instances (Sukhbaatar et al., 2015; Ratner et al., 2016; Jindal et al., 2016, 2019), or (2) develop techniques to filter noisy instruction instances (Swayamdipta et al., 2020; Yuan et al., 2024).

Unlike AeLMs, generative models (ArLMs) are pretrained on substantially larger datasets—often involving approximately 100 times more train-

[*]Corresponding author: maeda.hanafi@ibm.com
[†]Work done while at IBM Research


Instruction Template

Question: <question>

Output a 0 if the question is a non-standalone that is attached to readings of the sentences, otherwise output 1.

Question Examples

	Last_turn_question	Label	
✓	How much does it cost to visit?	Non-Standalone	✓
✓	What should I do if I smell burning insulation?	Non-Standalone	✗
✗	What time will it take to get a reply on an appeal?	Standalone	✓
✗	What is the court trial going to be held?	Non-Standalone	✗
	...		

Figure 1: An example of the noisy training dataset, containing both the *text noise* and the *label noise*. The phrase, "What time will it take...", is incorrect for expressing duration; "*How long* will it take..." is appropriate, and denotes *text noise*. The phrase, "What should I do...", can be answered without previous context, and denotes *label noise*. *Last_turn_question* denotes the current user utterance in multi-turn dialogue with an AI assistant.

ing tokens [1]—which enables them to serve as robust world models. This extensive pretraining allows ArLMs to learn new skills with fewer but higher-quality examples, whether through fine-tuning (Zhang et al., 2023) or in-context learning (Mavromatis et al., 2023). Given this capability, we aim to explore the following key questions in this work: (Q1) What is the impact of noisy instruction datasets on ArLMs acquiring a new skill? (Q2) Do noise mitigation techniques developed for AeLMs generalize to ArLMs? (Q3) How can we best identify and mitigate noise (both label noise and text noise) in instruction datasets for ArLMs?

Inspired by data cartography (Swayamdipta et al., 2020), which analyzes the training dynamics of AeLMs to identify noisy labels in fine-tuning datasets, we propose TDRANKER, a method that extends the cartography approach to autoregressive language models (ArLMs). We introduce a novel ranking function that allows us to extend proven techniques from data cartography on AeLMs to ArLMs. TDRANKER captures the training dynamics of ArLMs (GPT2 (Radford et al.)), ranks the instruction data samples based on the training dynamics, and categorizes the data samples into two broad groups: hard-to-learn and easy-to-learn. This classification helps pinpoint datapoints that may contain noise or require additional attention during the instruction fine-tuning process. Our ultimate goal in this paper is to develop a framework that helps quickly identify datapoints in need of human re-review from among thousands of noisy labels and nonsensical texts, improving the overall quality and reliability of instruction fine-tuning datasets. Our main contributions/observations are as follows:

- We propose TDRANKER– a data-driven approach that utilizes training dynamics during instruction fine-tuning to identify and remove both text noise and label noise in human-created datasets for ArLMs.
- We introduce a rank-by-correctness function categorizing datapoints from easy-to-learn to hard-to-learn, and demonstrating accumulation of significant noisy instances at the bottom of the ranking (§3.3).
- TDRANKER is effective across different models (GPT-2, BERT, LaMini-Cerebras-256M), demonstrating its robustness and **generalizability across both AeLMs and ArLMs** (§5.2).
- Iteratively applying TDRANKER reduces noise progressively. Ranking datapoints by correctness enables **at least 2x faster denoising** compared to other ranking functions (§5.3).
- TDRANKER is robust across datasets with different noise levels (10%, 30%, 50%, 70%), showing that it **consistently reduces noise** and improves data quality over multiple iterations (§5.4).
- Application to real-world datasets: We apply our method to real-world classification and generative tasks, identifying both text noise and label noise (§5.5).

The remainder of the paper is structured as follows: (§2) presents the literature review, followed by a detailed description of TDRANKER (§3) and the experimental setup (§4). We conclude with an in-depth analysis and discussion of the results (§5).

2 Related Works

Identifying and mitigating label noise in datasets is crucial for improving model generalization. Previous research has explored various approaches to detecting and handling noisy labels in both AeLMs and ArLMs. We categorize related work into two key areas: handling label noise in autoencoder-based models (discriminative) and addressing label noise in autoregressive models (generative).

[1] e.g., LLama vs BERT token comparison

Label Noise in Autoencoder-Based Models

Our work draws inspiration from data cartography (Swayamdipta et al., 2020), which maps datapoints based on their training dynamics to classify them into easy-to-learn, ambiguous, and hard-to-learn categories. The original data cartography framework was primarily designed for discriminative models, which capture confidence scores across epochs to determine the position of datapoints in a learned representation space. By computing variance-based features from these scores, prior work successfully identified datapoints that contributed to model uncertainty.

Beyond data cartography, other approaches have also aimed to identify and rectify noisy labels in discriminative models. For example, Reiss et al. (Reiss et al., 2020) identified annotation errors in the CoNLL 2003 dataset by leveraging an ensemble of models trained specifically for Named Entity Recognition (NER). However, such ensemble-based methods are model-specific and may not generalize well across different tasks and architectures. Our approach differs by being model-agnostic, applicable to both discriminative and generative frameworks without requiring specialized training or additional ensembles.

Another prominent line of research involves data programming (Ratner et al., 2016), which generates noisy labeled data and denoises it using a generative framework with a discriminative loss function. While effective, this method relies on human-curated labeling rules to synthesize noisy labels, limiting its adaptability to tasks where explicit labeling heuristics are unavailable. In contrast, our method leverages training dynamics to autonomously detect label noise without requiring human-crafted rules.

Label Noise in Autoregressive Models

Extending data cartography to autoregressive models presents unique challenges, as confidence scores from generative models are known to be unreliable without calibration (Ulmer et al., 2024; Guo et al., 2017). Unlike discriminative models, which produce well-defined probability distributions over fixed label sets, generative models generate free-form text, making their confidence scores harder to interpret.

Despite these challenges, researchers have explored strategies for identifying noise in generative datasets. The WANLI dataset (Liu et al., 2022) applied data cartography to human-annotated datasets, identifying difficult patterns and synthesizing new challenging examples for natural language inference. However, this work primarily focused on dataset augmentation rather than denoising.

Our approach builds upon these insights by applying a ranking function to training dynamics captured from autoregressive or autoencoder models. By ranking datapoints by their learning trajectories, specifically by using their correctness scores (§3.3), we effectively separate high-quality instances from noisy ones. Unlike prior methods that rely on ensemble-based heuristics or explicit rule-based labeling, our method generalizes across both discriminative and generative models without requiring manual intervention.

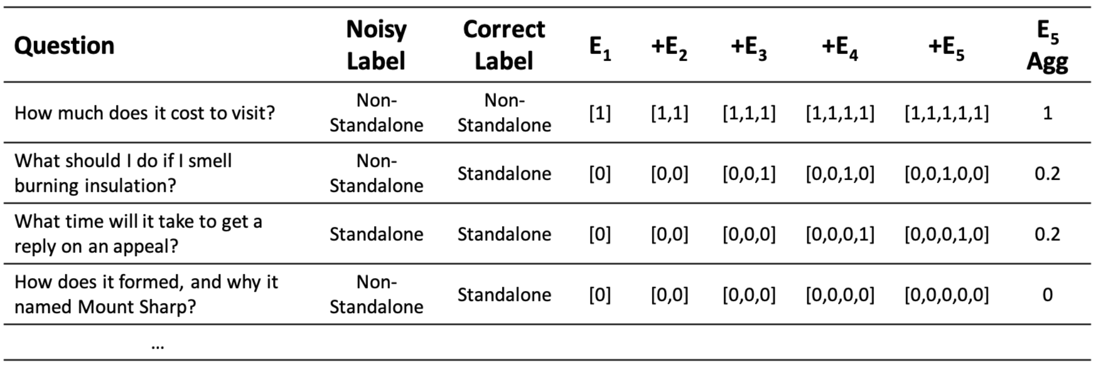
Overall, our work contributes to the ongoing effort of dataset refinement by providing a unified, model-agnostic method for detecting and filtering noisy labels across diverse learning paradigms.

3 Methodology

In this section, we describe our method, TDRanker, in detail. To identify noisy data in a dataset, the process involves two main steps. First, an ArLM like GPT-2 is trained on noisy data. During training, the model's behavior is tracked for each data point, including its predictions (adjusted to match the expected labels), whether the predictions are correct, and its confidence in the predictions (see §3.2). This information helps determine how easy/hard the model learns each data point. Second, the data points are ranked based on their learning difficulty, from easiest to hardest (see §3.3). Experiments show that data points that are harder to learn often have problems like incorrect labels or poor-quality text.

3.1 Task

Formally, we define an instruction-tuning **dataset** $D = \{(d_j, a_j) \mid j = 1, 2, \ldots, |D|\}$, where d_j is a **datapoint** in D and a_j is the **noisy label** (or **noisy text** if the task is generative) obtained from a human annotator that may or may not match the **actual output** l_j (not known *a priori*). A **task** is defined as an instruction for a model M to execute on a dataset D. For example, a task could involve classifying whether a given question is *standalone*, i.e., answerable without referring to the previous context in a multi-turn conversation. The outputs a_j and l_j are either labels or texts, based on whether the task is a classification or a generative one.



Question	Noisy Label	Correct Label	E_1	$+E_2$	$+E_3$	$+E_4$	$+E_5$	E_5 Agg
How much does it cost to visit?	Non-Standalone	Non-Standalone	[1]	[1,1]	[1,1,1]	[1,1,1,1]	[1,1,1,1,1]	1
What should I do if I smell burning insulation?	Non-Standalone	Standalone	[0]	[0,0]	[0,0,1]	[0,0,1,0]	[0,0,1,0,0]	0.2
What time will it take to get a reply on an appeal?	Standalone	Standalone	[0]	[0,0]	[0,0,0]	[0,0,0,1]	[0,0,0,1,0]	0.2
How does it formed, and why it named Mount Sharp?	Non-Standalone	Standalone	[0]	[0,0]	[0,0,0]	[0,0,0,0]	[0,0,0,0,0]	0
...								

Figure 2: Illustration of TDRANKER detecting noise in a small toy dataset over 5 epochs. The training dynamics are captured during instruction tuning (E_i denotes the training dynamic after the i'th epoch). After the 5th epoch, TDRANKER computes the mean of E_5, ranks datapoints, and identifies lower-ranking ones as likely noisy due to their harder-to-learn nature.

3.2 Capturing training dynamics

For a given dataset D relevant for a particular task, a **model** M is instruction-tuned over N epochs. We denote M_i to be the model after the i'th epoch. We also define the notion of a **generated label** (or **generated text** for generative task), y_{ij}, generated by the model M_i for a datapoint d_j.

We define **training dynamics** for each datapoint d_j. Intuitively, a training dynamic consists of characteristics regarding M_i for d_j, which includes:

- The confidence score, s_{ij}, represents the model confidence in generated output y_{ij} for datapoint d_j by model M_i at the i'th epoch.

- The correctness score, c_{ij}, represents the accuracy of the generated output y_{ij} for the data point d_j on a specific task performed by the model M_i. The calculation of c_{ij} is given by:

$$c_{ij} = \begin{cases} 1, & \text{if } y_{ij} \equiv a_j \\ 0, & \text{if } y_{ij} \not\equiv a_j \end{cases}$$

If $M_i(d_j) \equiv a_j$, then the annotator-provided label or text matches the model M_i's prediction. We denote S the set of all confidence scores and C the set of all correctness scores. We denote t_{ij} as the training dynamic for datapoint d_j and model M_i, where $t_{ij} = (s_{ij}, c_{ij})$, $t_{ij} \in T$, and T is the set of all the training dynamics for D and M.

3.3 Ranking datapoints by training dynamics

For each datapoint d_j, we either aggregate the model confidence score (s_{ij}) or the correctness scores (c_{ij}) collected across all N epochs. We denote this aggregation function as $R(d_j) : d_j \rightarrow \mathbb{R}$. Once aggregated, all $\{R(d_1), R(d_2), \ldots, R(d_{|D|})\}$ are then arranged in ascending order, categorized from easy-to-learn to hard-to-learn examples. Top-ranked being easy-to-learn and bottom-ranked being hard-to-learn.

In data cartography (Swayamdipta et al., 2020), M was an AeLM (e.g. BERT), T consisted of only S, and datapoints were ranked according to the confidence scores in S. Specifically the following ranking functions were used:

- Ranking datapoints in D by the mean **confidence scores** across the epochs:

$$R(d_j)_S = \mu_{s_j} = \frac{1}{N}\sum_{i=1}^{N} s_{ij}$$

- Ranking datapoints in D by the **variability of the confidence scores**:

$$R(d_j)_V = -\sigma_{s_j} = -\sqrt{\frac{1}{N}\sum_{i=1}^{N}(s_{ij} - \mu_{s_j})^2}$$

If one were to rank the datapoints d_j by the confidence scores using $R(d_j)_V$, the higher-ranking datapoints have smaller variability in their confidence scores than the lower-ranking datapoints that have higher variability in their confidence scores[2].

Unlike data cartography, TDRANKER works on M that can either be an ArLM or an AeLM to detect noise within the instruction tuning dataset. As highlighted by Ulmer et al. (2024), the confidence scores S produced by ArLMs are often unreliable. To address this, TDRANKER ranks data

[2]Appendix A describes how the ranking functions in data cartography groups the data based on different characteristics.

points based on C, which quantifies the correctness of the model's predictions over multiple training epochs. Specifically, we define the aggregation function $R(d_j)_C$ for correctness by computing the **average correctness score** across all epochs for each datapoint d_j:

$$R(d_j)_C = \mu_{c_j} = \frac{1}{N}\sum_{i=1}^{N} c_{ij}$$

Our experiments demonstrate that ranking data points based on correctness scores and examining those with lower rankings enables effective identification of potentially noisy data points d_j. These instances often contain label and text noise, which may necessitate re-evaluation by a human annotator or, in certain situations, removal from the dataset to maintain data integrity.

4 Experiments

This section describes experimental settings.

4.1 Ranking Functions

We conducted experiments with various ranking functions to identify data points for cleaning by selecting those with the lowest ranks: (1) correctness scores average, $R(d_j)_C$, which is our novel ranking function, (2) confidence scores[3] variability, $R(d_j)_V$, and (3) confidence scores average, $R(d_j)_S$.

4.2 Selection Methods

We compare TDRanker against a baseline approach *Random* where a random percentage of data points was selected for cleaning.

4.3 Datasets

We conducted experiments on five datasets, comprising three public datasets and two enterprise datasets[4] (real-world dataset).

4.3.1 Public Datasets

We used three public datasets: (1) QNLI (Wang et al., 2018), (2) OpenBookQA (Mihaylov et al., 2018), and MELD (Poria et al., 2019). These datasets cover a variety of tasks (see Appendix C). Because these are all classification datasets, we introduced a controlled level of noise by flipping a percentage of the labels. A percentage of noise was introduced into the train *labels*, where for each label, an equal portion of noise was introduced, e.g. QNLI has 5% noise for each label, $\{0, 1\}$, resulting in 10% label noise. We categorize all of these datasets as *classification datasets*. To understand how our approach differs from the data cartography work on AeLMs and extend those ideas to ArLMs, we conducted experiments for each dataset on two models: an ArLM, GPT-2, and an AeLM, BERT. Both models were fine-tuned for 20 epochs, during which the training dynamics were recorded.

[3]All reported confidence scores from ArLMs are calibrated using the histogram binning method (Guo et al., 2017).

[4]Sufficient details on the enterprise datasets are provided in the paper for clarity.

Dataset	Domain	Type	Task	# Dp
Gov-Stand	Gov	C	Standalone	600
Gov-Pert	Gov	C	Pertinence	799
Ent-Stand	Cloud	C	Standalone	2200
Gov-Qw	Gov	G	Query Rewrite	300
Ent-Qw	Cloud	G	Query Rewrite	1100

Table 1: Real-world datasets. No groundtruth data was available aside from the annotator-given labels and annotator-given texts that we presume to be noisy. Gov and Cloud were transformed into 6 datasets for fine-tuning GPT-2 on a variety of tasks, types (C for classification, G for generative), and dataset sizes (#Dp).

Task	Type	Description
Standalone task	C	Given a conversation, classify whether the last question depends on the previous part of the conversation such as through co-reference or ellipsis, i.e. a standalone question.
Pertinence task	C	Given a conversation, classify whether the last response is pertinent to the last question. In our particular scenario, pertinence takes on a stricter definition; if a substantial part of the agent's response does not answer the primary question posed by the human, the response is considered non-pertinent, even in cases where the response may still be on topic.
Query rewrite task	G	Given a conversation, rewrite a standalone version of the last question in the conversation such that the standalone version can be answered without the context in the conversation.

Table 2: Tasks on real-world datasets. C for classification, G for generative task.

4.3.2 Real-World Datasets

After confirming the effectiveness of TDRanker for identifying subsets of noisy data using ArLMs, we tested our method on two real-world datasets, which we call *Gov* and *Cloud*. These datasets are multi-turn conversations with an AI agent over two domains (cloud documentation of a major cloud provider and web pages from the government domain). The datasets were created and annotated by an external annotation service. Labels that describe the type of turn (i.e. pertinent/non-pertinent question, standalone/non-standalone question, etc.) were also given. We asked annotators to write the standalone version of each question in the conversation between a human and an agent such that the standalone question can be answered without the previous context. For experimentation, the two datasets were transformed into several fine-tuning datasets as shown in Table 1. Six *classification tasks* and *generative tasks*, which we call Gov-Stand, Gov-Pert, Ent-Stand, Gov-Qw, and Ent-Qw, were formed for each of the datasets based on the available labels, as shown in Table 2. The datasets were then used to fine-tune GPT-2 over 20 epochs. During fine-tuning, the training dynamics were captured after each epoch.

5 Results

In this section, we evaluate different aspects of our proposed method: the impact of ranking functions on data selection (§5.1), the impact of different model choices (§5.2), the effectiveness of iterative application for dataset denoising (§5.4), the impact of different noise levels (§5.3), and the applicability of our method to real-world noisy datasets (§5.5).

5.1 Impact of Ranking Functions

For each dataset, we analyzed the training dynamics obtained from fine-tuning on the noisy dataset and ranked the datapoints using three different ranking functions and two selection methods. We then created a cleaner dataset by automatically cleaning the bottom 10% datapoints output by the respective ranking method. We then fine-tuned the model for an additional 20 epochs on the cleaner dataset (see Table 3).

For GPT-2, we observe that prioritizing the removal of hard-to-learn datapoints—identified by ranking them based on correctness scores $R(i, d_j)_C$ and TDRANKER selection method—led to a consistent performance improvement across all the datasets as compared to random selection. This is because, with GPT-2, correctness scores cluster the noisy data towards the bottom of the ranked list. Similar consistent gains are observed with BERT model across all the classification datasets[5].

[5]Figure 7 in Appendix E provides ranking clusters for different ranking functions.

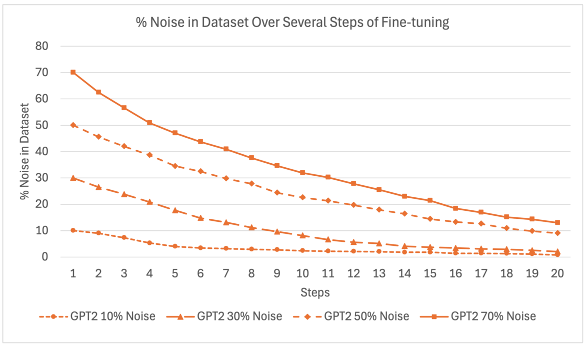


Figure 3: Impact of Noise Levels: We iteratively denoise QNLI using our method by fine-tuning GPT-2 and then ranking the datapoints by correctness.

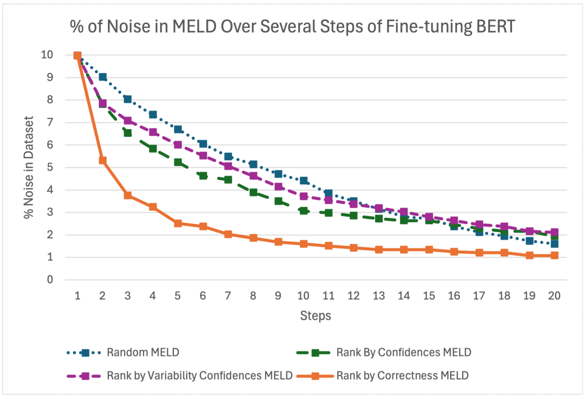


Figure 4: % noise in the MELD dataset over several steps. Ranking by correctness denoises the dataset over fewer steps than the other ranking functions.

TDRANKER with correctness scores shows consistent gains across most datasets for both models.

5.2 Model-Agnostic Denoising

To assess whether TDRANKER is agnostic to the choice of the denoising model, we experimented with model interchangeability. Specifically, we denoised the dataset using one model (e.g., GPT-2) ranked by correctness and then fine-tuned the other model (e.g., BERT) on the cleaned data for 20 epochs. Denoising was conducted over 20 steps, with each step consisting of five training epochs. As shown in Table 4, **TDRANKER effectively cleans noise using correctness scores, regardless of the denoising model employed.**

5.3 Iterative Denoising Across Models

We experimented with iterative dataset denoising by applying our method over multiple *steps*. In each step, we: (1) fine-tuned GPT-2 or BERT for

Dataset	Step	Ranking Function	Selection Method	GPT2				BERT			
				% Cln	Acc	P	R	% Cln	Acc	P	R
QNLI	1	None	None	0	0.72	0.77	0.67	0	0.72	0.75	0.72
	2	None	Random	8%	0.73	0.77	0.70	15%	0.77	0.77	0.77
		$R(d_j)_V$	TDRanker	9%	0.72	0.79	0.65	14%	0.77	0.78	0.77
		$R(d_j)_S$	TDRanker	2%	0.74	0.81	0.66	15%	0.75	0.77	0.75
		$R(d_j)_C$	TDRanker	**34%**	**0.79**	**0.84**	**0.75**	**29%**	**0.78**	**0.79**	**0.78**
OPQA	1	None	None	0	0.27	0.27	0.27	0	0.67	0.67	0.67
	2	None	Random	4.44%	0.47	0.47	0.47	5.65%	0.66	0.66	0.66
		$R(d_j)_V$	TDRanker	6.67%	0.51	0.51	0.51	**5.85%**	0.68	0.68	0.68
		$R(d_j)_S$	TDRanker	8.48%	0.51	0.51	0.51	5.45%	0.67	0.67	0.67
		$R(d_j)_C$	TDRanker	**11.31%**	0.50	0.50	0.50	5.25%	**0.69**	**0.69**	**0.69**
MELD	1	None	None	0	0.69	0.63	0.72	0	0.76	0.77	0.76
	2	None	Random	8.65%	0.72	0.67	0.73	16.45%	0.75	0.75	0.75
		$R(d_j)_V$	TDRanker	4.76%	0.74	0.69	0.77	16.45%	0.75	0.75	0.75
		$R(d_j)_S$	TDRanker	14.72%	0.74	0.69	0.76	20.35%	0.74	0.75	0.74
		$R(d_j)_C$	TDRanker	**41.55%**	**0.76**	**0.72**	0.77	**48.48%**	**0.78**	**0.78**	**0.78**

Table 3: Impact of ranking functions. Step 1 and 2 shows the model's performance after fine-tuning the model on the noisy and on the cleaner version of the noisy dataset, respectively. *Selection method* indicates how the datapoints are selected to clean. Random means we randomly selected 10% of the dataset to clean, while TDRanker means we selected 10% of the bottom-ranked datapoints. % Cln denotes the percent of the bottom-ranked datapoints that were cleaned. More detailed visualization of the impact of ranking can be found in the Appendix, Figure 6 and 7.

Dataset	% Noise	Denoising Model	Model FT	Fine-tuned Model		
				Acc	P	R
QNLI	10%	BERT	GPT2	0.79	0.81	0.79
		GPT2	GPT2	0.79	0.84	0.75
		BERT	BERT	0.78	0.79	0.78
		GPT2	BERT	0.76	0.78	0.76
OPQA	5%	BERT	GPT2	0.50	0.50	0.50
		GPT2	GPT2	0.49	0.49	0.49
		BERT	BERT	0.69	0.69	0.69
		GPT2	BERT	0.64	0.64	0.64
MELD	10%	BERT	GPT2	0.75	0.70	0.75
		GPT2	GPT2	0.76	0.72	0.77
		BERT	BERT	0.78	0.78	0.78
		GPT2	BERT	0.75	0.76	0.75

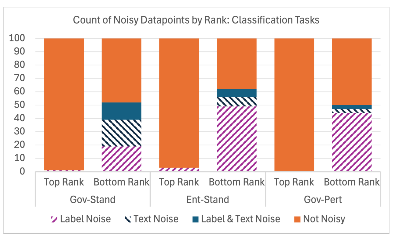
Table 4: Model-Agnostic Denoising: Denoising Model de-noises the dataset and another model is fine-tuned on the denoised dataset.

five epochs on the noisy dataset to capture training dynamics, (2) ranked datapoints using different ranking functions, and (3) cleaned the bottom 10% of datapoints to simulate human review and relabeling. The process was repeated iteratively with the progressively cleaner dataset.

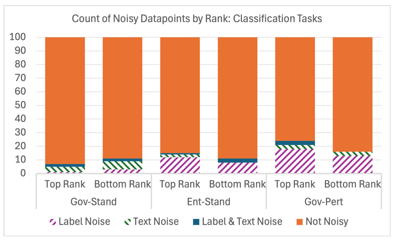
We observe that with each *step* each ranking function helps reduce the noise in the datasets. However, rank by correctness cleans the noise twice as fast as other ranking functions (see Figure 4). More plots showcasing similar results for both models (GPT-2 and BERT) on the QNLI and MELD dataset can be seen in Figure 8 in Appendix F. This proves that **ranking datapoints by correctness results in faster denoising compared to other ranking functions,** with the effect being particularly pronounced for BERT.

5.4 Impact of Noise Levels

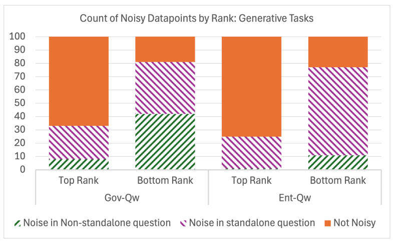
To evaluate the robustness of our method, we experimented with varying noise levels (30%, 50%, and 70%) in the dataset and analyzed how denoising progressed over multiple steps. Figure 3 illustrates the reduction in noise as our method iteratively fine-tunes a GPT-2 model, ranks datapoints by correctness scores, and removes the bottom 10% of low-ranking datapoints in each step. We further replicated this experiment using a smaller model, LaMini-Cerebras-256M (Wu et al., 2023), and BERT, observing a similar denoising trend (Figure 9 in Appendix G). These results confirm that **TDRanker effectively reduces noise across different models with varying noise levels**, demonstrating its generalizability and effectiveness in dataset refinement.



(a) $R(d_j)_C$ on classification tasks.



(b) $R(d_j)_V$ on classification tasks.



(c) $R(d_j)_C$ on generative tasks.

Figure 5: Identifying noise in real-world datasets using TDRANKER on GPT-2 training dynamics.

5.5 Analysis on Real-world Datasets

We benchmarked TDRANKER on the 6 real-world datasets for both the classification tasks and the generative tasks as described in Section 4.3.2. For the real-world datasets, ground-truth data was not available. A researcher manually reviewed the bottom-ranking datapoints, identifying ones requiring human review or relabeling. For each datapoint in the classification task datasets, we marked whether the noise was in the text, label, or both. For each datapoint in the generative task datasets, we marked whether the noise was in the non-standalone or the standalone question that the annotators created.

5.5.1 Results on Classification Tasks

Ranking by correctness helped identify more noisy datapoints than ranking by the variability of the confidence scores. Ranking by correctness in the Gov-Stand dataset resulted in a total of 52 noisy datapoints in the bottom 100 ranking datapoints as opposed to only 1 noisy datapoint in the top 100 ranking datapoints (Figure 5a). Across all three real-world datasets, ranking by correctness results in a clustering of noisy datapoints towards the bottom of the rank.

While label noise dominated both Ent-Stand and Gov-Pert datasets, there was more text noise in the Gov-Stand dataset; twenty datapoints contained only text noise, while 13 datapoints contained both text noise and label noise. We discovered that much of the noise introduced in the texts was linked to particular annotators for which English was not their first language.

5.5.2 Results on Generative Tasks

We also experimented with generative tasks, specifically the query rewrite tasks on the real-world datasets shown in Table 1. We captured the training dynamics from fine-tuning GPT-2 to rewrite a query into a standalone question. To calculate correctness and quantify how much GPT-2's generated output matches the annotator's standalone question, we used an LLM Judge[6], specifically LLaMA-3-70B. Datapoints that the model struggled with in earlier epochs but then finally learned are considered easy to learn, and thus, we calculate the correctness score based on the last ten epochs instead of all 20 epochs.

In the hard-to-learn datapoints, text noise in the non-standalone question almost always cascades into the standalone version of the question, since (as part of the task) annotators are instructed to write the standalone version based on the non-standalone question. Our method can cluster more datapoints with this type of fatal[7] noise towards the bottom ranks for human re-review (see the count of noise in non-standalone questions in the bottom ranks in Figure 5c). In the Gov-Qw dataset, there was 5% more fatal, text noise in the non-standalone questions in the bottom rank than in the top rank. Ultimately, **TDRANKER enabled the discovery of fatal noise in our real-world datasets**.

6 Conclusion

In this paper, we propose an approach for identifying subsets of data containing a high percentage of noise, including label noise and text noise. Training dynamics are captured during the fine-tuning stage and are used to rank datapoints from easy-to-learn and hard-to-learn, where hard-to-learn datapoints often contain a higher percentage of noise. Our experiments show that this approach can be used regardless of whether the model is an AeLM (BERT) and ArLM (GPT-2, LaMini) architecture. Our approach identified subsets of noise in real-world datasets, with the noisier subsets containing

[6]See more in Appendix D

[7]For instance, in a conversation about compost and compost bins, a non-standalone question follows it, "And what conditions should maintain?". The annotator's question has grammatical errors making it unclear what "it" refers to, i.e. the conditions of the compost bin or the compost itself.

more fatal noise that cascades into other attributes of the data.

Limitations

For the purposes of quickly identifying subsets of noise in our human-created datasets, our approach has been experimented with small language models. The results may not generalize to larger language models. Moreover, while the annotator-created datasets contain noise, the researcher re-reviewing the datapoints that are part of the subset of data identified as likely to be noisy by our approach is also subject to human error. Our experiments have only been conducted on text-based datasets and conversational datasets and may not generalize to datasets of other modalities.

Ethical Concerns

Some of the datasets used in our experiments are open-sourced and their usage is permitted as long as their original work is cited. The data created from the real-world datasets were from annotators from a dedicated external annotation service that disburses payment to annotators. Before data creation, annotators consent to its collection and usage. Because our work also demonstrated the feasibility of identifying noise in the annotator's created data, we do not report annotator characteristics, as doing so adjacent to discussions of identifying subsets of errors in the created data may jeopardize the annotator's jobs. Company policy dictates how data is handled and stored.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

AI@Meta. 2024. Llama 3 model card.

Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. 2017. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. 2024. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1321–1330. JMLR.org.

Ishan Jindal, Matthew Nokleby, and Xuewen Chen. 2016. Learning deep networks from noisy labels with dropout regularization. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 967–972. IEEE.

Ishan Jindal, Daniel Pressel, Brian Lester, and Matthew Nokleby. 2019. An effective label noise model for DNN text classification. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3246–3256, Minneapolis, Minnesota. Association for Computational Linguistics.

Alisa Liu, Swabha Swayamdipta, Noah A. Smith, and Yejin Choi. 2022. WANLI: worker and AI collaboration for natural language inference dataset creation. *CoRR*, abs/2201.05955.

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *CoRR*, abs/1711.05101.

Costas Mavromatis, Balasubramaniam Srinivasan, Zhengyuan Shen, Jiani Zhang, Huzefa Rangwala, Christos Faloutsos, and George Karypis. 2023. Which examples to annotate for in-context learning? towards effective and efficient selection. *arXiv preprint arXiv:2310.20046*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *EMNLP*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Diane Oyen, Michal Kucer, Nicolas Hengartner, and Har Simrat Singh. 2022. Robustness to label noise depends on the shape of the noise distribution. *Advances in Neural Information Processing Systems*, 35:35645–35656.

Soujanya Poria, Devamanyu Hazarika, Navonil Majumder, Gautam Naik, Erik Cambria, and Rada Mihalcea. 2019. MELD: A multimodal multi-party dataset for emotion recognition in conversations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 527–536, Florence, Italy. Association for Computational Linguistics.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Frederick Reiss, Hong Xu, Bryan Cutler, Karthik Muthuraman, and Zachary Eichenberger. 2020. Identifying incorrect labels in the CoNLL-2003 corpus. In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 215–226, Online. Association for Computational Linguistics.

Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. 2023. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2015. Training convolutional networks with noisy labels. In *3rd International Conference on Learning Representations, ICLR 2015*.

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.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Dennis Ulmer, Martin Gubri, Hwaran Lee, Sangdoo Yun, and Seong Oh. 2024. Calibrating large language models using their generations only. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15440–15459, Bangkok, Thailand. Association for Computational Linguistics.

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

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Minghao Wu, Abdul Waheed, Chiyu Zhang, Muhammad Abdul-Mageed, and Alham Fikri Aji. 2023. Lamini-lm: A diverse herd of distilled models from large-scale instructions. *CoRR*, abs/2304.14402.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Bo Yuan, Yulin Chen, Yin Zhang, and Wei Jiang. 2024. Hide and seek in noise labels: Noise-robust collaborative active learning with LLMs-powered assistance. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10977–11011, Bangkok, Thailand. Association for Computational Linguistics.

Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792*.

Yivan Zhang, Gang Niu, and Masashi Sugiyama. 2021. Learning noise transition matrix from only noisy labels via total variation regularization. In *International Conference on Machine Learning*, pages 12501–12512. PMLR.

Dawei Zhu, Michael A Hedderich, Fangzhou Zhai, David Ifeoluwa Adelani, and Dietrich Klakow. 2022. Is bert robust to label noise? a study on learning with noisy labels in text classification. *arXiv preprint arXiv:2204.09371*.

A Data Cartography

In data cartography, experiments on training dynamics using an AeLM M, specifically BERT, demonstrate that ranking data points by their confidence scores, $R(d_j)_S$, effectively differentiates between data points that are consistently correct across epochs (higher correctness scores) and those that are less likely to be correct (lower correctness scores).

In other words, ranking by $R(d_j)_C$ or $R(d_j)_S$ indicates that higher-ranking data points are easier for the model to learn, whereas lower-ranking data points are more challenging to learn. Additionally, an auxiliary ranking function based on the *variability of the confidence scores*, $R(d_j)_V$, distinguishes ambiguous data points from non-ambiguous ones. Harder-to-learn and ambiguous data points are more likely to have lower correctness scores compared to easier-to-learn data points.

While in (Swayamdipta et al., 2020), identifying hard-to-learn datapoints meant identifying out-of-distribution datapoints, in our experiments, we show that hard-to-learn datapoints indicate a higher likelihood of datapoints that were noisy. In other words, noisy datapoints identified using our method grouped together datapoints that the model struggled to learn (hard-to-learn datapoints) and that the model did not learn well (ambiguous datapoints).

B Training & Hyperparameters

Each experiment is performed on a single NVIDIA Tesla V100 GPU 32 GB. Our implementation uses the Huggingface Transformers library (Wolf et al., 2019). For training BERT, the same scheduler and optimizer in the experiments for data cartography (Swayamdipta et al., 2020) were also used in our experiments for BERT. For training the ArLMs, a learning rate of 2e-4 was used alongside AdamW optimizer (Loshchilov and Hutter, 2017).

C Pulic Dataset

D LLM Judge Prompts

As part of calculating the *correctness score* of a given datapoint for a generative task such as query rewrite, we use an LLM Judge to identify how similar a model's generated output matches the annotator's golden standalone question. We prompted LLaMA-3-70B model as shown in Listing 1[8].

E Impact of Ranking Functions

A close-up on the rankings of the QNLI dataset from the training dynamics obtained from GPT2 shows that ranking by correctness scores clusters the noisy data towards the bottom (see Figure 6b), unlike ranking by variability of confidence scores (see Figure 6a). Given the noisy datapoints identified in the bottom rank, we applied a step of denoising on the dataset, fine-tuning the model on a cleaner dataset, and then ranked the datapoints based on the correctness scores. We visualize the rankings after a round of denoising in Figure 6c, where once again, the bottom ranks cluster more noisy datapoints compared to the top ranks.

Figure 7 provides ranking clusters for different ranking functions. Similar consistent gains are observed with BERT model across all the classification datasets. This shows that TDRanker with correctness scores shows consistent gains across all datasets for both the models.

F Iterative Denoising

We experimented with denoising datasets by iteratively applying our method for several *steps*.We fine-tuned GPT2 and BERT for several steps, where in each step we (1) fine-tune the model over 5 epochs with the noisy dataset to obtain the training dynamics, (2) rank the datapoints based on different ranking functions and then denoise the bottom 10% of the datapoints to simulate human review/relabeling of the bottom-ranking datapoints. We then use the cleaner dataset and repeat the step several times. In Figure 8, ranking the datapoints by correctness denoises the dataset over fewer steps compared to the other ranking functions.

G Impact of Noise Levels

We experimented with different levels of noise (30%, 50%, 70%) in the QNLI dataset to see how our method denoises the dataset as it progresses through the steps. We denoised the dataset using our method by fine-tuning 3 different models: (1) GPT-2, (2) a similarly smaller model, LaMini-Cerebras-256M (Wu et al., 2023), and (3) BERT. Figure 9 shows the percentage of noise as we iteratively denoise the dataset using our method by fine-tuning a LaMini and BERT model over several steps (the GPT-2 results are shown in Figure 3). After each step, we rank the datapoints by correctness scores to denoise the bottom 10% ranking datapoints. TDRanker effectively reduces noise across different models with varying noise levels.

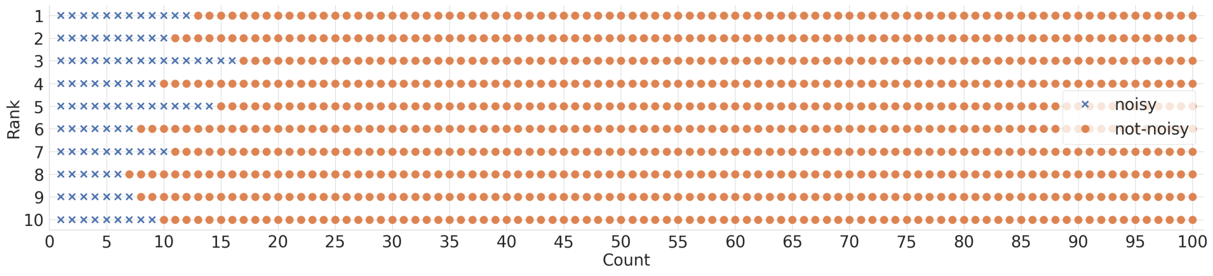
[8]Prompts and LLM Judges were from IBM's Ecosystem Engineering team

Dataset	Task	Description	Train	Test	N%
QNLI	Answerability	Given a group of sentences and a question, “output 0 if the group of sentences contains the information required to answer the question; otherwise, output 1”.	1100	200	10%
OpenBookQA	Multiple-choice question-answering	Given a sentence stem and multiple-choice options, “select the correct answer for the multiple-choice question: A, B, C, or D.”	4957	500	5%
MELD	Emotion recognition	Given a conversation and a subsequent utterance, “determine the emotional tone of the utterance: anger or surprise. Output 0 for anger and 1 for surprise.”	2314	626	10%

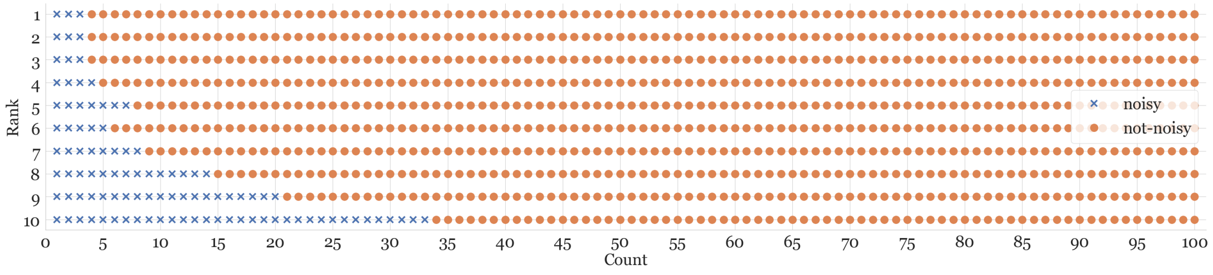
Table 5: Tasks on public datasets and statistics, with % noise (N%)

Listing 1: Prompt for correctness score

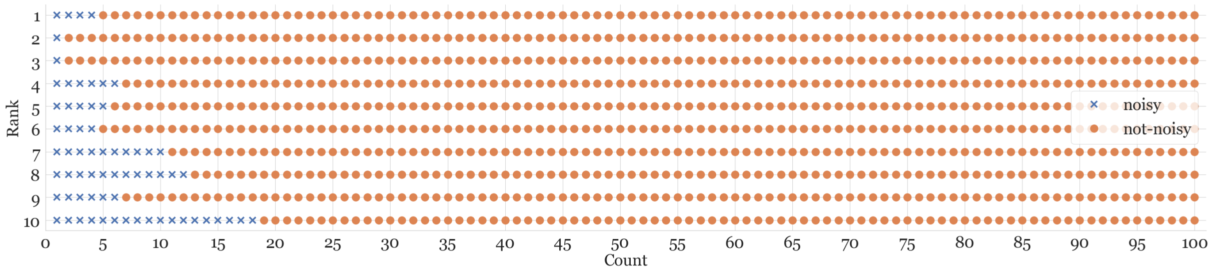
```
Follow these below structured steps to accurately assess query transformations and
    ensure alignment with provided criteria.
1. **Role and Task**: Assume the role of an impartial assistant and judge. Your task
     is to evaluate query transformations using provided information. You will
    receive a Conversation History, Follow Up Query, Golden Rewritten Query, and a
    Rewritten Query for evaluation.
2. **Initial Setup**: Begin by reviewing the Conversation History to understand the
    context. Then, introduce the Follow Up Query that requires transformation.
3. **Golden Rewritten Query**: Examine the Golden Rewritten Query, which serves as
    the correct reference for adding context to the Follow Up Query based on the
    entities from the Conversation History, if necessary. Ensure that the Golden
    Rewritten Query is fully correct and comprehensive.
4. **Evaluation Criteria**: Evaluate the Rewritten Query based on the following
    criteria:
   - Output {{"Grade": "1"}} if the Rewritten Query matches the Golden Rewritten
       Query in terms of entities and intents and with the Conversation History.
   - Output {{"Grade": "0"}} if the Rewritten Query contains additional information
       not present in the Golden Rewritten Query.
   - Output {{"Grade": "0"}} if the Rewritten Query is missing information that is
       present in the Golden Rewritten Query.
5. **Output Format**: Format your evaluation output strictly as {{"Grade": "
    evaluated grade"}} to ensure clarity and consistency in assessment.

Input:
Conversation History: What is the SSI for Best Buy?
The Supplier Stability Index (SSI) for Best Buy is 10. This indicates that Best Buy
    has a high likelihood of experiencing significant financial or operational
    instability over the next 3 months. This could manifest as the company ceasing
    operations, seeking legal relief from creditors, going into receivership or
    reorganization, making arrangements for the benefit of creditors, or becoming
    inactive due to merger or acquisition related activity.
Follow Up Query: Do they have any government indicators?
Golden Rewritten Query: Does Best Buy have any government indicators?
Rewritten Query: Does Best Buy have any government indicators?

Output:
{{"Grade": "1"}}

Input:
Conversation History: What is the website for Adobe?
The website for Adobe is www.adobe.com.
Follow Up Query: What is the SSI for Pfizer?
Golden Rewritten Query: What is the SSI for Pfizer?
Rewritten Query: What is the website for Pfizer?

Output:
{{"Grade": "0"}}

Input:
Conversation History: {prompt_parameter_1}
Follow Up Query: {prompt_parameter_2}
Golden Rewritten Query: {prompt_parameter_3}
Rewritten Query: {prompt_parameter_4}

Output:
```



(a) **Ranking datapoints by variability of confidence scores.** The rank of each datapoint is obtained after (1) one round of fine-tuning the model on the noisy QNLI dataset, and then (2) ranking the datapoints by their variability of confidence scores. Ranking by variability of confidence scores does not cluster the noisy datapoints towards the bottom of the plot. However, ranking by correctness scores clusters the noisy datapoints towards the bottom of the plot (see Figure 6b).

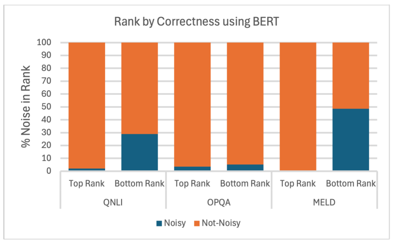


(b) **Ranking datapoints by correctness scores.** The rank of each datapoint is obtained after (1) one round of fine-tuning the model on the noisy QNLI dataset, and then (2) ranking the datapoints by their correctness scores. Higher rank indicates that they are considered easier to learn than lower-ranking datapoints (hard-to-learn). Note the noisy datapoints cluster towards the bottom of the plot.

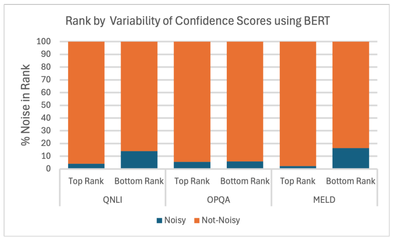


(c) **Ranking datapoints by correctness scores after cleaning the bottom 10% of the dataset** identified in Figure 6b. The rank of each datapoint is obtained after (1) one round of fine-tuning the model on the noisy QNLI dataset, (2) ranking the datapoints by their correctness scores, (3) cleaning the bottom 10% of the ranked dataset, (4) a second round of fine-tuning the model on the partially cleaned dataset, and (5) ranking the datapoints by correctness scores. In short, we applied an additional round of our method: plot the rank of the dataset after fine-tuning the model on the cleaner dataset.

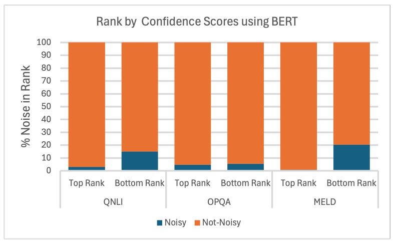
Figure 6: Plots of the QNLI dataset. Each point in the plot is a datapoint, from the the QNLI dataset (size 1000) with 100 synthetically-created noisy labels. Each point on the plot is marked as noisy or not.



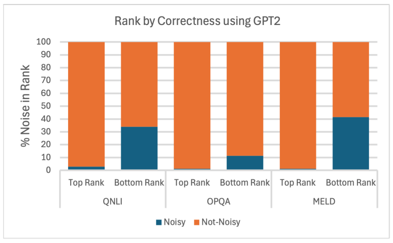
(a) Rank datapoints by correctness from training dynamics obtained by fine-tuning BERT.



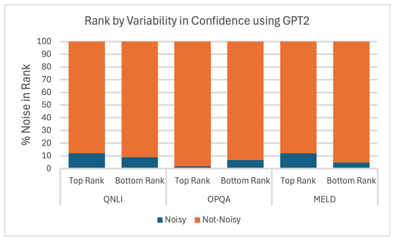
(b) Rank datapoints by variability of confidence scores from training dynamics obtained by fine-tuning BERT.



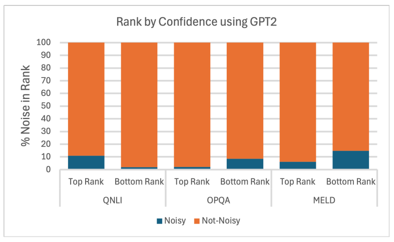
(c) Rank datapoints by confidence scores from training dynamics obtained by fine-tuning BERT.



(d) Rank datapoints by correctness from training dynamics obtained by fine-tuning GPT2.

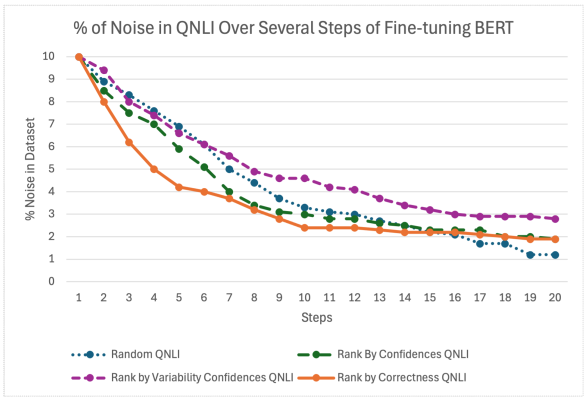


(e) Rank datapoints by variability of confidence from training dynamics obtained by fine-tuning GPT2.

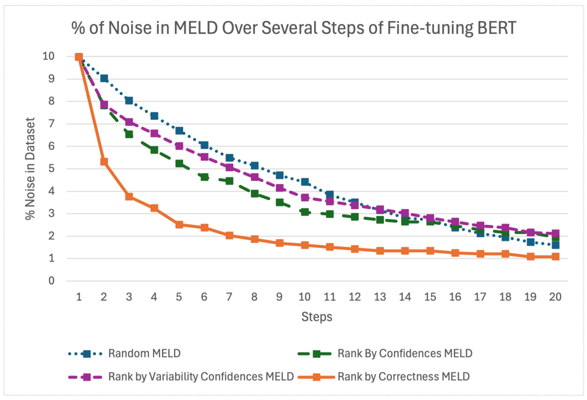


(f) Rank datapoints by confidence from training dynamics obtained by fine-tuning GPT2.

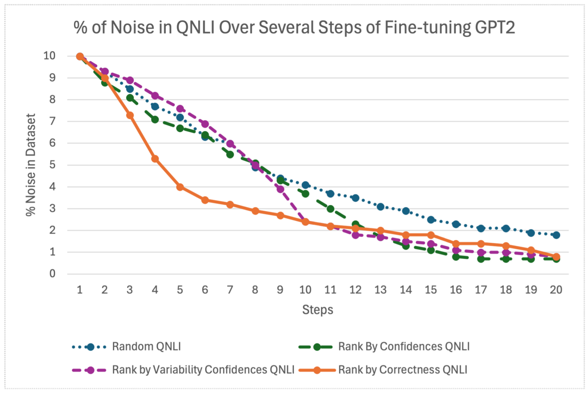
Figure 7: Comparing different ranking functions on the training dynamics obtained by fine-tuning a model on the public datasets. Top rank refers to the top-ranking 10% of the dataset and bottom rank refers to the bottom-ranking 10% of the dataset. We report the percentage of noise within the rank. Note that ranking by correctness, $R(d_j)_C$, clusters the noisy datapoints towards the bottom ranking datapoints as opposed to ranking by variability in confidence scores, $R(d_j)_V$.



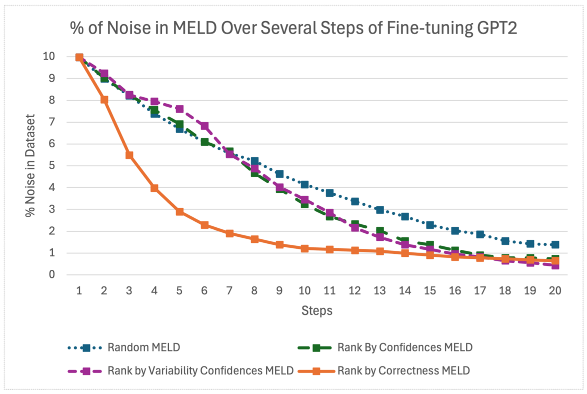
(a) % noise in the QNLI dataset from fine-tuning BERT.



(b) % noise in the MELD dataset from fine-tuning BERT.



(c) % noise in the QNLI dataset from fine-tuning GPT2.



(d) % noise in the MELD dataset from fine-tuning GPT2.

Figure 8: Percent of noise in the QNLI and MELD dataset over several steps. Both datasets had a 10% of noise introduced into the dataset. Ranking the datapoints by correctness denoises the dataset over fewer steps than the other ranking functions.

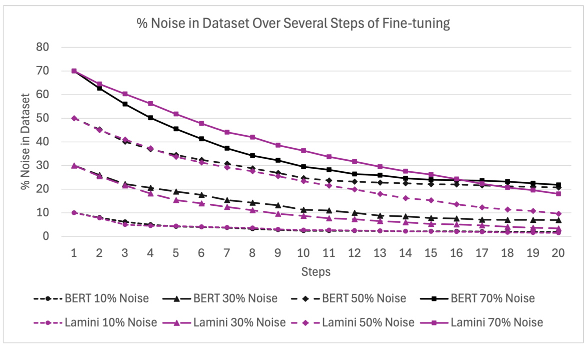


Figure 9: Iteratively denoising the QNLI dataset using our method (capturing training dynamics by fine-tuning BERT and LaMini and then ranking the datapoints by correctness). Different levels of noise are experimented with.