DEFT-UCS: Data Efficient Fine-Tuning for Pre-Trained Language Models via Unsupervised Core-Set Selection for Text-Editing

Devleena Das[*]
Georgia Institute of Technology
ddas41@gatech.edu

Vivek Khetan
Accenture Labs
vivek.a.khetan@accenture.com

Abstract

Recent advances have led to the availability of many pre-trained language models (PLMs); however, a question that remains is how much data is truly needed to fine-tune PLMs for downstream tasks? In this work, we introduce DEFT-UCS, a data-efficient fine-tuning framework that leverages unsupervised core-set selection to identify a smaller, representative dataset to fine-tune PLMs for text-generation needed for text editing tasks such as simplification, grammar correction, clarity, etc. We examine the efficacy of DEFT-UCS across multiple text-editing tasks, and compare to the state-of-the art text-editing model, CoEDIT. Our results demonstrate that DEFT-UCS models are just as accurate as CoEDIT, across eight different datasets consisting of six different editing tasks, while finetuned on 70% less data.

1 Introduction

How much data do we need to fine-tune a pre-trained language model (PLM) for a specific downstream task? While successes in language modelling have led to numerous publicly available PLMs and ability to produce fine-tuned models for downstream tasks - the answer mostly remains, "as large as possible, and of good quality". For example, Alpaca, an instruction-following model, is trained with 52k data samples (Taori et al., 2023). Similarly, CoPoet, a collaborative poetry writing system is fine-tuned using 87k data samples (Chakrabarty et al., 2022). MetaMath, a math-reasoning LLM is fine-tuned with 395k data samples (Yu et al., 2023). Although fine-tuning PLMs on specific task results in performance gain, acquiring large amounts of data for fine-tuning is not easy for real-world applications which often require niche knowledge and domain expertise.

Researchers have explored variety of methods primarily focused on improving the computational efficiency of fine-tuning, including parameter-efficient fine-tuning approaches (PEFT) to reduce computational costs by optimizing parameter updates (Fu et al., 2023; Hu et al., 2021) as well as leveraging active-learning for iteratively selecting data samples during training (Su et al., 2022; Diao et al., 2023). Instead, our work focuses on improving the *data efficiency* of PLM fine-tuning without requiring iterative fine-tuning. Specifically, we explore how to fine-tune PLMs with significantly less data samples and without a cost to model performance. Related to language models, researchers have experimented with different core-set selection metrics (Paul et al., 2021; Sorscher et al., 2022) to improve the data efficiency during *pre-training*. Marion et al. (2023) demonstrated how perplexity, L2-Error Norm (EL2N) and memorization can be utilized to select smaller, good quality datasets for pre-training. Similarly, (Attendu and Corbeil, 2023) leverage EL2N to dynamically remove data samples with high EL2N between training epochs. However, these metrics assume access to task data and reference models to perform dataset pruning. In real world applications, utilizing such supervised, data-pruning metrics are less realistic since large amounts of annotated task-specific data may be costly to acquire. This leads us to our main research question: *How can we leverage unsupervised data pruning to fine-tune PLMs for downstream tasks in a more data efficient manner?*

In this work, we introduce a new data-efficient fine-tuning framework, DEFT-UCS, that uses unsupervised core-set selection to minimize the amount of labelled data needed to fine-tune PLMs for text generation related to multiple text-editing tasks. Our framework is inspired by (Sorscher et al., 2022), who utilize clustering-based dataset pruning to reduce training samples for image-classification models, and to the best of our knowledge, our framework is the first to leverage unsupervised core-set selection for data-efficient fine-tuning of

[*]Contributed during an internship at Accenture Labs, SF

PLMs.

We study the utility of DEFT-UCS in fine-tuning PLMs for text-generation across eight different datasets consisting of six different text-editing tasks, and compare DEFT-UCS models to the state-of-the-art text-editing model, CoEDIT(Raheja et al., 2023). Our contributions are as follows:

- We introduce DEFT-UCS, a data-efficient-fine tuning framework that leverages unsupervised core-set selection via clustering to identify a smaller representative set of data needed to fine-tune PLMs.
- We show that DEFT-UCS, utilizing only 32.5% of CoEDIT's training data, is able to produce fine-tuned models with improved accuracy on four different text-editing tasks, and similar accuracy on two text-editing tasks compared to CoEDIT (Raheja et al., 2023).
- We performed a human evaluation with 3 evaluators to assess the quality of text-edits from our DEFT-UCS model. Evaluators found edits generated by DEFT-UCS model as similar or preferred over CoEDIT (Raheja et al., 2023).

2 Related Works

Efficient Fine-Tuning of LLMs Most work on efficient fine-tuning techniques for LLMs have primarily focused on parameter-efficient fine-tuning (PEFT) approaches (Fu et al., 2023; Hu et al., 2021), improving computation efficiency by updating a subset of model parameters. Recently, there has been an increasing focus on improving the data-efficiency of LLMs, considering how to pre-train and fine-tune LLMs with smaller subsets of data (Zhou et al., 2023a; Mukherjee et al., 2023; Chen et al., 2023; Marion et al., 2023; Attendu and Corbeil, 2023; Ivison et al., 2022). For instance, Zhou et al. (2023a) introduce LIMA, an approach to fine-tune LLaMA (Touvron et al., 2023) with only 1k diverse and high quality samples. However, the LIMA approach is underspecificed without a general subsampling procedure. Also, Chen et al. (2023) develop Skill-It!, which creates efficient datasets by learning hierarchical relationships between samples. However, identifying hierarchical relationships is non-trivial and not all datasets may include them. More closely related to our work, Ivison et al. (2022) leverage K-Nearest Neighbors to learn multiple data-efficient fine-tuned models for individual tasks. Instead, we aim to learn a single data-efficient fine-tuned model that performs competitively across a variety of datasets. Similarly, Marion et al. (2023) utilize perplexity and EL2N, to find smaller datasets for LLM pre-training, and Attendu and Corbeil (2023) uses EL2N to iteratively remove unimportant samples during fine-tuning. Both Marion et al. (2023) and Attendu and Corbeil (2023) assume access to task data to train various reference models for few epochs to calculate EL2N and perplexity. In contrast, we leverage unsupervised core-set selection, omitting the need for any reference model during the dataset sampling step.

Core-Set Selection & Dataset Distillation Several works in ML have developed variety of core-set selection (Har-Peled and Kushal, 2005) and dataset pruning (Paul et al., 2021) methods to find smaller subsets of data needed to train deep learning models without model performance loss. CRAIG (Mirzasoleiman et al., 2020) finds core-sets by approximating gradient calculations, while RETRIEVE (Killamsetty et al., 2021) finds core-sets by optimizing for model loss. Also, Yang et al. (2022) utilize Influence Functions (Koh and Liang, 2017) to prune redundant samples. A unifying idea among these methods is the need for labelled data.

Alternatively, core-set selection methods for unlabelled data have used clustering methods. Birodkar et al. (2019) use Agglomerative clustering to find semantic similarities among data points and prune redundant samples. Similarly, Sorscher et al. (2022) use vanilla k-means clustering and distances to cluster centroids for pruning easy and hard samples. Recently, data distillation algorithms have also been developed to improve data-efficient model training (Zhou et al., 2023b). Typically, data distillation methods generate new synthetic datasets in which data samples are edited to preserve more information for performance generalization (Lei and Tao, 2023). Our work considers efficient core-set selection without the generation of a synthetic dataset. Specifically, our work extends (Sorscher et al., 2022), which performs easy and hard sampling to reduce training data from a single dataset, ImageNet (Deng et al., 2009). In our work, we additionally consider the effects of random sampling (mix of easy and hard), access to initial seed data, and study the effects of such sampling techniques for dataset pruning across several text-editing tasks.

Instruction Tuning for Text-Editing Recently, Instruction tuning of PLMs has shown impressive success in its ability to enable PLMs to follow instructions as well as improvement in generalization across various tasks in zero/few shot settings (Min et al., 2022; Wei et al., 2021). Training models to explicitly follow natural language instructions has become increasingly popular for text-editing tasks as well. Shu et al. (2023) develop RewriteLM by fine-tuning PaLM (Chowdhery et al., 2022) variants for the task of rewriting long-form texts. Similarly, Schick et al. (2022) develop PEER by fine-tuning T5 (Raffel et al., 2020) variants to emulate the collaborative writing process. Additionally, Raheja et al. (2023) develop CoEDIT by fine-tuning Flan T5 (Chung et al., 2022) models to perform single and compositional edits across multiple edit tasks. Furthermore, Zhang et al. (2023) produce an instruction-tuned LLaMA model that improve text-editing capabilities. A commonality across these works include the usage of large scale datasets for fine-tuning. For example, CoEDIT (Raheja et al., 2023) and Zhang et al. (2023) leverage datasets with 82k and 60k examples, respectively. In our work, DEFT-UCS maximizes model performance of fine-tuned models in a *data efficient* manner by finding a representative, smaller dataset needed for fine-tuning. We investigate the efficacy of our DEFT-UCS framework to instruction fine-tune PLMs for eight text-editing tasks.

3 Problem Formulation

We formulate DEFT-UCS as an unsupervised core-set selection problem (Sorscher et al., 2022) in contrast to existing dataset pruning methods which primarily use supervised core-set selection (Attendu and Corbeil, 2023; Marion et al., 2023).

Specifically, let D represent an existing large dataset, P represent a PLM, and M_D represent P fine-tuned on D. Our DEFT-UCS framework aims to find a representative core-set $D_c \subset D$ such that leveraging D_c can fine-tune P and result in a fine-tuned model M_{D_c} with comparable performance to M_D. Note, we refer to comparable evaluation performance in the form of both quantitative NLP metrics and qualitative human evaluations. Specific to unsupervised core-set selection, DEFT-UCS finds D_c without needing D to include annotations or labels. Thus, we find D_c by only using the input samples $\{x_1..x_n\}$ within D. These input samples, in the context of instruction fine-tuning, represent task instructions and input texts.

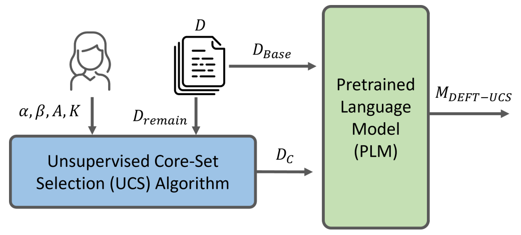

D
D_{Base}
α, β, A, K
D_{remain}
Unsupervised Core-Set Selection (UCS) Algorithm
D_C
Pretrained Language Model (PLM)
$M_{DEFT-UCS}$

Figure 1: Our DEFT-UCS framework utilizes unsupervised core-set selection (UCS) to find a core-set of data D_c, as well as initial seed data, D_{base} to produce a fine-tuned PLM, $M_{DEFT-UCS}$.

To perform unsupervised core-set selection, we build upon the SoTA clustering-based core-set selection method by Sorscher et al. (2022), given its extensive evaluations against other supervised-based core-set selection methods. While Sorscher et al. (2022) demonstrate the efficacy of clustering-based core-set selection for ImageNet (Deng et al., 2009), our work is the first to investigate the effectiveness of clustering-based core-set selection in non-classification tasks, such as fine-tuning PLMs for multiple text-editing tasks.

Algorithm 1 Unsupervised Core-set Selection (UCS)

Input: $D_{remain} = \{x_0, x_1...x_n\}$ - Large Dataset
Input: K - Num. of Clusters
Input: A - Amount of samples per cluster
Input: α, β, - Sampling Weights
Output: $D_c = \{x_j..x_p\}$ - Core-Set

```
 1: $D_c = \emptyset$
 2: $D_{embed}$ = ComputeEmbedding($D_{remain}$)
 3: $Cl_{1:K}, Ce_{1:K}$ = KMeans($D_{embed}$, $K$)
 4: for $i$ in $K$ do
 5:     for $d$ in $Cl_i$ do
 6:         $dist_{list}$ = StoreCosineDistance($d$, $Ce_i$)
 7:     end for
 8:     $dist_{sorted}$ = sort($dist_{list}$)
 9:     $D_{sampled}$ = $dist_{sorted}$[0 : $\alpha$*A]
            + $dist_{sorted}$[ -$\beta$*A:]
10:     $D_c$ = updateCoreSet($D_{sampled}$, $D_c$)
11: end for
12: return $D_c$
```

4 DEFT-UCS Framework

Figure 1 outlines our DEFT-UCS framework which leverages *unsupervised, clustering-based core-set* *selection* (*UCS*) to find a subset of D that fine-tunes a PLM without compromising model performance. We consider a scenario in which there exists an initial amount of data, $D_{base} \subset D$, that is sampled in a stratified manner to provide an overall representation of the downstream fine-tuning task. Let D_{remain} represent the remaining data after D_{base} is sampled. The goal of UCS is to then find a core-set $D_c \subset D_{remain}$ that enriches D_{base} such that D_c and D_{base}, together, form a representative subset that can be used to fine-tune a PLM and result in a fine-tuned model $M_{DEFT-UCS}$ with comparable performance to M_D, a PLM fine-tuned with D. In Algorithm 1, we detail the crux of our DEFT-UCS framework, the UCS method.

4.1 Clustering in UCS

The first step in UCS includes transforming D_{remain} into a meaningful embedding representation D_{embed}. UCS clusters D based on its latent-space representation, using previously learned embedding spaces, such as sentenceBert (Reimers and Gurevych, 2019). Choosing an appropriate embedding representation is important, given that such representation impacts the downstream clustering task within UCS. In Section 5, we detail the types of learned embedding spaces we evaluate and the best embedding representation found for encoding sentence-based datasets.

Given D_{embed}, we perform K-Means clustering to separate D_{embed} into K clusters. Note, the value of K is dependent on D, and defining K requires domain knowledge about the dataset to understand the different categories or tasks represented in D. Alternatively, K can be automatically derived using metrics such as Silhouette Score (Shahapure and Nicholas, 2020). The resulting K clusters, $Cl_{1:K}$, and cluster centroids, $Ce_{1:K}$, are utilized to compute the cosine distance between each data sample d in a cluster Cl_i, and corresponding centroid Ce_i.

4.2 Sampling D_c in UCS

We leverage the clustering categorization presented in Sorscher et al. (2022) to sample D_c from D_{remain}. Specifically, Sorscher et al. (2022) explain that data samples can be categorized as “easy” or “hard” examples. In the context of unsupervised clustering, Sorscher et al. (2022) leverage a data sample’s distance to its cluster centroid to define easy and hard samples. Therefore, easy/hard samples within a cluster are those closest/furthest to the cluster centroid. Given such definition, in UCS, we retrieve a weighted sampling of easy and hard samples from each cluster, denoted as $D_{sampled}$. The α and β weights control the distribution of easy and hard samples in $D_{sampled}$, and A represents the total number of samples retrieved per cluster.

Note, D_{base}, K, A, α, and β are hyperparameters within DEFT-UCS, manually set by domain-experts. Given this is the first work, to our knowledge, to propose data-efficient fine-tuning for PLMs leveraging UCS, we perform an exhaustive investigation on how these hyperparameters influence fine-tuning performance (see Section 7). Future work includes investigating automatic selection of such hyperparameters.

5 DEFT-UCS Applied to Text-Editing

We evaluate the utility of DEFT-UCS in the context of instruction-based fine-tuning for multiple text editing tasks. To our knowledge, the current SoTA instruction fine-tuned text-editing LM is CoEDIT (M_{CoEDIT})[1] trained on dataset D_{CoEDIT} (Raheja et al., 2023). Overall, D_{CoEDIT} includes 82k good-quality edit instructions spanning six different edit-tasks (Raheja et al., 2023) (D_{CoEDIT} detailed in Appendix A.1). Given the data quality in D_{CoEDIT} and SoTA performance of M_{CoEDIT}, we apply DEFT-UCS to D_{CoEDIT}. Below, we detail the hyper-parameter choices in DEFT-UCS in the context of D_{CoEDIT}.

5.1 D_{Base} in CoEDIT

Recall D_{Base} refers to initial data sampled in a stratified manner used for fine-tuning. In our work, stratified sampling is performed based on the different tasks represented in D. During our evaluations, we study how the size of D_{Base} may influence hyperparameter selection within our UCS algorithm for producing a well-performing $M_{DEFT-UCS}$. In the context of CoEDIT, we experiment with $D_{Base} = \{10\%, 20\%, ..80\%\}$, representing 10% to 80% of D_{CoEDIT}. Note, D_{CoEDIT} is a fully annotated dataset; however, when performing core-set selection $D_c \subset D$, we only consider the input sentences.

5.2 DEFT-UCS Hyperparameters

Given that D_{CoEDIT} includes seven edit-intentions, we set $K = 7$, allowing the K-Means Clustering within UCS to separate D_{remain} into 7 clusters. Additionally, recall from Sec. 4

[1]https://github.com/vipulraheja/coedit

that α and β represent the sampling weights for extracting easy and hard data samples from each cluster to form $D_{sampled}$. To understand the upper and lower bound effects of α and β, we study three variants of $D_{sampled}$, representing three different sampling types: $D^{hard}_{sampled}$, $D^{easy}_{sampled}$ and $D^{rand}_{sampled}$. Specifically, $D^{hard}_{sampled}$ is represented by $\alpha = 0$ and $\beta = 1.0$, $D^{easy}_{sampled}$ is represented by $\alpha = 1.0$ and $\beta = 0$, and $D^{rand}_{sampled}$ approximates $\alpha = 0.5$ and $\beta = 0.5$, denoting random samples extracted per cluster. We also experiment with sampling different amounts of data from each cluster, denoted by $A = \{285, 570, 857\}$. Such settings of A approximate $\{2000, 4000, 6000\}$ total samples from D_{remain} respectively, and represent $\{2.5\%, 5\%, 7.5\%\}$ percent of D_{remain}.

5.3 Dataset Embedding

Recall that the UCS algorithm in DEFT-UCS performs clustering using a learned embedding representation of the input data samples. We investigate several embedding representations and select the best embedding representation by its ability to inform accurate clusters. Specifically, we study sentence-level encodings from Sentence-T5 (Ni et al., 2021), BART (Lewis et al., 2019) CLS token embeddings, as well as averaged word token embeddings from Flan-T5 (Chung et al., 2022). From an ablation study, our results demonstrate that leveraging Sentence-T5 (Ni et al., 2021) results in the best K-Means Clustering performance. The ablation study results are in Appendix B.

5.4 Model Fine-Tuning

Raheja et al. (2023) develop CoEDIT-Large, CoEDIT-xl, and CoEDIT-xxl by fine-tuning Flan-T5's Large, XL and XXL models, respectively. In our work, we focus our comparisons against CoEDIT-Large, referred to as M_{CoEDIT}. Therefore, in our framework, we fine-tune Flan-T5-Large, producing $M^{Flan-T5-LG}_{DEFT-UCS}$. Details on our fine-tuning implementation are in Appendix A.2.

6 Experiments

6.1 Evaluation Datasets

Table 1 presents eight test datasets used in our evaluation. We performed evaluations across six different edit tasks including simplification, coherence, clarity, fluency, grammar correction and neutralization improvement. See Appendix C for dataset

Evaluation Dataset	Edit Task
TurkCorpus (Xu et al., 2016a)	Simplification
Asset (Alva-Manchego et al., 2020)	Simplification
Iterator Coherence (Du et al., 2022)	Coherence
Iterator Clarity (Du et al., 2022)	Clarity
Iterator Fluency (Du et al., 2022)	Fluency
Iterator Global (Du et al., 2022)	Clarity, Coherence, Fluency
JFLEG (Napoles et al., 2017)	Grammar Correction
WNC (Pryzant et al., 2020)	Neutralization

Table 1: A list of datasets, spanning six editing tasks, on which we evaluate our DEFT-UCS models.

details. For fair comparisons, these datasets include the publicly available datasets evaluated by CoEDIT (Raheja et al., 2023), and are present in several text-editing benchmarks, including EDITEVAL (Dwivedi-Yu et al., 2022).

6.2 Metrics

We examine SARI (Xu et al., 2016b) and ROUGE-L (Lin, 2004) scores for our quantitative evaluations. SARI scores are also utilized in prior text-editing tasks (Raheja et al., 2023). During our human evaluation, we analyze users' perceived accuracy percentage (*PA%*), which measures the percent of times users select a text-editing model for producing accurately edited sentences.

6.3 Baselines

We compare our fine-tuned models via DEFT-UCS, $M_{DEFT-UCS}$, to the following baselines.

CoEDIT-Large The primary baseline of our work is the original CoEDIT-Large model (Raheja et al., 2023), M_{CoEDIT}, which uses the entire 82k samples in D_{CoEDIT} to fine-tune Flan-T5 Large. To compare against M_{CoEDIT}, we utilize the released CoEDIT model[2] and compare SARI and ROUGE-L scores for each evaluation dataset.

LIMA Approach We also compare our DEFT-UCS method to the LIMA approach (Zhou et al., 2023a). Following the LIMA approach of using high quality and diverse 1k data points, we select 1k data samples via stratified random sampling from D_{CoEDIT} for fine-tuning Flan-T5. We refer to such LIMA-inspired model as M_{LIMA}. Prior work by Raheja et al. (Raheja et al., 2023) validate the high-quality data samples in D_{CoEDIT}, and stratified random sampling ensures data diversity, allowing all editing tasks within D_{CoEDIT} to be equally represented.

[2]https://huggingface.co/grammarly/coedit-large

Non-Instruction Fine-Tuned LLMs We also compare our $M_{DEFT-UCS}$ with LLamA2-7B ($M_{LLAMA2-7B}$) (Touvron et al., 2023), Flan-T5-Large ($M_{FLAN-T5-LG}$) (Chung et al., 2022) and BLOOM-560M ($M_{BLOOM-560M}$) (Scao et al., 2022), in Zero-Shot settings, to understand how $M_{DEFT-UCS}$ compares to non-instruction fine-tuned LLMs.

7 Results

Our results below show that DEFT-UCS can provide a *data-efficient* method for producing competitive fine-tuned models for six different text-editing tasks.

7.1 DEFT-UCS vs. CoEDIT

Figure 2 shows that our DEFT-UCS framework generates fine-tuned models with comparable performance to M_{CoEDIT} in terms of SARI (Fig. 2a) and Rouge-L (Fig. 2b) scores, using lower fractions of D_{CoEDIT}. These results indicate that unsupervised core-set selection within DEFT-UCS can effectively find a D_c for fine-tuning without compromising downstream task performance.

The DEFT-UCS models in Figure 2 reflect the existence of *a* competitive DEFT-UCS model, and depending on the evaluated text-editing task, a different fraction of D_{CoEDIT} results in the *most* competitive performances. For example, to achieve comparable performance on the WNC dataset for the neutralization task, a DEFT-UCS model needs above 80% of D_{CoEDIT}. In contrast, for the Asset dataset and simplification task, around 12% of D_{CoEDIT} is needed to surpass M_{CoEDIT} SARI and ROUGE-L scores. We hypothesize that subjectivity in the neutralization task (WNC) increases the complexity of the data samples and more data is required to fine-tune a competitive model in comparison to less subjective editing tasks such as, text-simplification (Asset). Interestingly, between datasets for the same editing task (Asset, Turk), we notice differences in the fraction of D_{CoEDIT} needed for competitive DEFT-UCS models.

7.2 DEFT-UCS vs. LIMA Approach

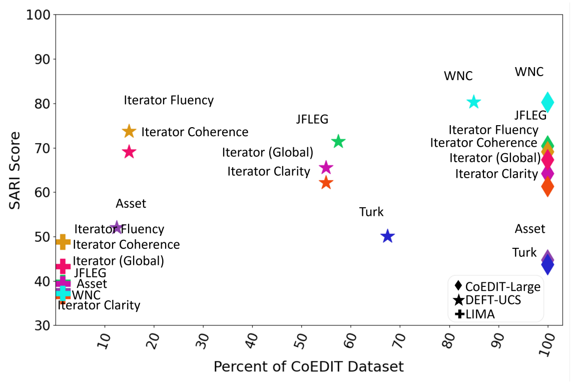
We observe across all evaluation tasks, M_{LIMA} has lower SARI and ROUGE-L scores compared to M_{CoEDIT} and our DEFT-UCS models. These results show that the LIMA (Zhou et al., 2023a) approach may not be generalizable to domain-specific LM tasks such as text-editing and more experimentation is needed to understand its limitations. Moreover, these results indicate that smarter sampling techniques that go beyond data quality and diversity are needed for competitive model performances, such as considering distance metrics in embedding spaces as utilized in DEFT-UCS.

7.3 Overall DEFT-UCS Model

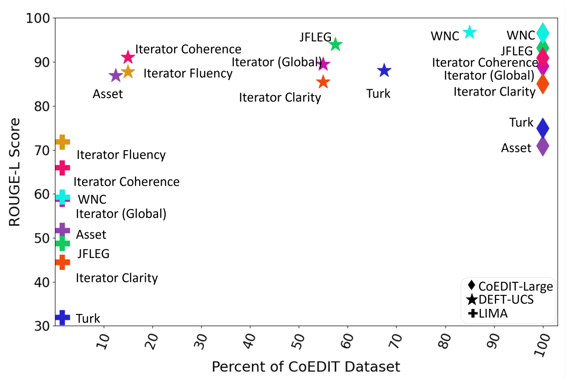
In Section 7.1, we found that the most competitive DEFT-UCS model for each evaluation dataset uses a different fraction of D_{CoEDIT}. Therefore, we performed an additional analysis to study which combination of hyper-parameters result in an overall best-performing DEFT-UCS model, one that achieves or surpasses M_{CoEDIT} performances on most evaluation datasets using a much smaller fraction of D_{CoEDIT}. Fig. 3(a) and Fig. 3(b) show that fine-tuning Flan-T5 Large with only 32.5% of D_{CoEDIT} and performing hard sampling ($\alpha = 0$, $\beta = 1.0$), results in the best overall DEFT-UCS model, $M_{DEFT-UCS}^{FLAN-T5-LG}$, surpassing M_{CoEDIT} SARI and ROUGE-L scores on six of the eight evaluation datasets. Overall, 32.5% represents the smallest fraction of D_{CoEDIT} that results in competitive SARI and ROUGE-L scores on most evaluation datasets.

Note, 32.5% of D_{CoEDIT} is composed of D_{base}, initial data available for fine-tuning, and D_c, the output of UCS within DEFT-UCS. In the context of $M_{DEFT-UCS}^{FLAN-T5-LG}$, D_{base} is a stratified 30% subset from D_{CoEDIT}, and D_c is composed of another 2.5% of D_{remain} ($A = 2000$ samples per cluster) retrieved from UCS by performing hard sampling.

Model Performance Table 2 shows the SARI and ROUGE-L scores of our best DEFT-UCS model, $M_{DEFT-UCS}^{FLAN-T5-LG}$, fine-tuned with only 32.5% of D_{CoEDIT}. We find that $M_{DEFT-UCS}^{FLAN-T5}$ performs better than M_{LIMA} and $M_{FLAN-T5-LG}$ on four datasets, and comparably on two datasets, WNC and JFLEG. Note, the exploration of whether DEFT-UCS produced models can significantly outperform M_{CoEDIT} requires in-depth evaluations across multiple NLP tasks, and is an important future work. Most importantly, our results highlight the utility of DEFT-UCS by showing that a much smaller fraction of D_{CoEDIT}, can be used to produce a comparable fine-tuned text-editing model. We also observe that $M_{LLAMA2-7B}$ and $M_{BLOOM-560}$ have much lower ROUGE-L scores compared to all other models. After examining model generated outputs, we see



(a)



(b)

Figure 2: Comparisons between the CoEDIT model (Raheja et al., 2023), LIMA-inspired model M_{LIMA} (Zhou et al., 2023a), and our DEFT-UCS models with respect to SARI (a) and ROUGE-L (b) scores.

Models	**Turk**	**Asset**	**Iterator Coherence**	**Iterator Clarity**	**Iterator Fluency**	**Iterator Global**	**JFLEG**	**WNC**
$M_{DEFT-UCS}^{Flan-T5-LG}$	**46.6 / 81.1**	**46.8 / 76.9**	**68.9** / 90.9	**61.8 / 85.3**	**69.9 / 96.9**	**64.7 / 89.1**	70.2 / 93.1	79.0 / 96.5
M_{CoEDIT}	43.7 / 74.9	44.7 / 70.9	67.3 / **91.1**	61.3 / 85.1	69.1 / 96.6	64.2 / 89.0	**70.4 / 93.2**	**80.2 / 96.5**
M_{LIMA}	23.8 / 31.9	37.8 / 51.7	43.3 / 65.9	36.5 / 55.5	48.8 / 71.9	39.4 / 58.9	39.7 / 48.8	37.2 / 59.3
$M_{LLAMA2-7B}$	36.8 / 17.3	41.6 / 20.3	35.8 / 26.2	41.2 / 28.5	40.4/ 33.8	38.3/ 29.7	46.0 / 17.0	27.3 / 17.2
$M_{FLAN-T5-LG}$	32.3 / 59.1	41.3 / 74.7	36.7 / 52.4	34.3 / 54.3	37.9 / 64.9	35.5 / 57.7	51.3 / 80.9	30.7 / 48.9
$M_{BLOOM-560M}$	27.3 / 7.7	32.0 / 8.2	19.1 / 8.8	20.6 / 9.7	16.3/ 8.2	19.6 / 9.5	27.9 / 4.9	18.8/ 8.1

Table 2: Comparisons between the overall best DEFT-UCS model, $M_{DEFT-UCS}^{FLan-T5-LG}$ with all other baselines, with the first value representing SARI score and second value representing ROUGE-L score. Note, scores for LLAMA-7B and BLOOM-560 model (Zero-shot) generations are calculated by first removing the prepended input sequence.

that lower ROUGE-L scores are attributed to long, repeated sentences from $M_{LLAMA2-7B}$ $M_{BLOOM-560}$. Appendix D.2 provides example edited sentences from each model.

Influence of D_{Base} & Sampling Methods Based on downstream tasks, the amount of D_{Base} may vary. Thus, we analyze how the size of D_{base} may influence the sampling method utilized in DEFT-UCS for producing best-performing models. Figure 4 summarizes the win percentages among the three sampling methods (random sampling, easy sampling, hard sampling) as the size of D_{base} increases. Win percentage is defined as the percent of times a particular sampling method achieves the highest SARI (Fig. 4a) or ROUGE-L (Fig. 4b) score across all evaluation datasets. From Figure 4a and Figure 4b, we observe that as D_{Base} increases, even across different D_c amounts, random sampling results in better SARI and ROUGE-L performances compared to easy and hard sampling. However, with lower amounts of D_{Base}, hard sampling results in better performance. We hypothesize that with lower amounts of D_{Base}, sampling harder examples may allow the model to generalize to unseen examples. Interactions between D_{Base} and sampling type may be dataset and task dependent, and future work should experiment with these hypotheses for different task-specific applications.

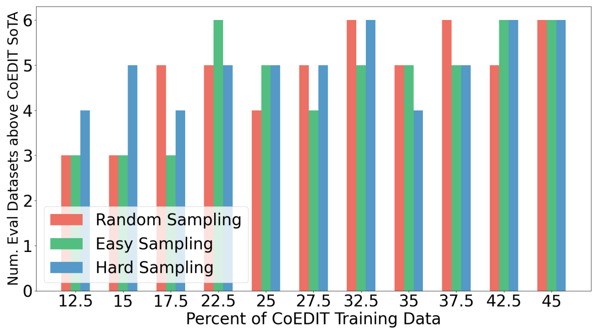
Model	**Perceived Accuracy (*PA%*)**
$M_{DEFT-UCS}^{Flan-T5-LG}$	83.8 %
M_{CoEDIT} (Raheja et al., 2023)	70.5%

Table 3: Perceived accuracy from human evaluation.

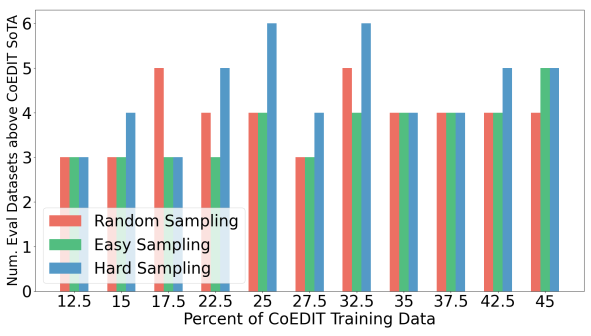
7.4 Human Evaluation

We hired three computer scientists with English as their primary language for our human evaluation. We created a human-eval test set by randomly sampling 35 examples from seven text-editing dataset in Table 1.[3] For each sample in the human-eval test set, evaluators were provided two edited sentence generated using $M_{DEFT-UCS}^{FLAN-T5-LG}$ and M_{CoEDIT}. Evaluators were then asked to select the most accurately edited sentence. Given that many edited sentences from $M_{DEFT-UCS}^{FLAN-T5-LG}$ and M_{CoEDIT} were

[3]We did not sample from Iterator Global since such dataset is a combination of Iterator Clarity, Fluency and Coherence.

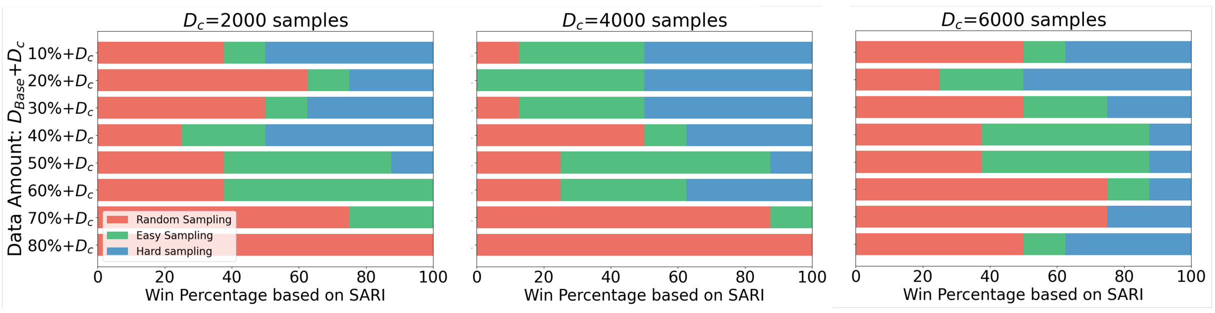


(a)

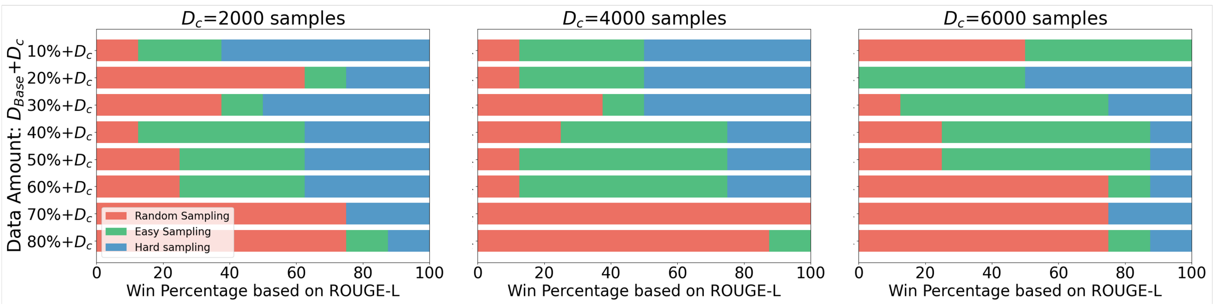


(b)

Figure 3: Utilizing hard sampling in UCS results in a best, overall DEFT-UCS model that requires only 32.5% of D_{CoEDIT} to beat 6/8 evaluation datasets considering SARI (a) and ROUGE-L (b) scores.



(a)



(b)

Figure 4: With less D_{base}, leveraging hard sampling in our DEFT-UCS leads to better performing models (winning %); as D_{base} increases, random sampling leads to better performing models.

similar or identical, evaluators were able to select more than one edited-sentence as accurately edited. To reduce bias, the generated sentence ordering from the models was randomized.

Table 3 summarizes the average perceived accuracy percentages (*PA%*). Overall, our $M_{DEFT-UCS}^{FLAN-T5-LG}$ results in higher PA% compared to M_{CoEDIT}. We also calculated the inter-rater reliability score to understand the agreement among evaluators on their PA%, and found moderate agreement with a Fleiss-Kappa (Fleiss and Cohen, 1973) score of 0.44. These results indicate that evaluators perceived our $M_{DEFT-UCS}^{FLAN-T5-LG}$ to produce accurately edited-sentences with comparable quality between M_{CoEDIT} and $M_{DEFT-UCS}^{FLAN-T5-LG}$.

8 Conclusion

We introduce DEFT-UCS, a data-efficient fine-tuning framework that leverages unsupervised core-set selection to find the minimum amount of data needed to fine-tune a PLM for text-editing tasks. Our best performing DEFT-UCS model, fine-tuned with only 32.5% of the CoEDIT dataset (Raheja et al., 2023), has comparable performance to the SoTA CoEDIT (Raheja et al., 2023) on two text-editing tasks, and improved performance to CoEDIT (Raheja et al., 2023) on four text-editing tasks, and the LIMA approach (Zhou et al., 2023a). Human evaluators also preferred edits generated by DEFF-UCS model over CoEDIT.

These results show the overall utility of our DEFT-UCS framework towards data-efficient fine-tuning of PMLs in text-editing tasks. To better understand the generalizability of DEFT-UCS, we plan to first apply it to more text-generation tasks in the future. Subsequently, we aim to benchmark the efficacy of different data-sampling strategies across various PLMs for these tasks.

Limitations First, the hyper-parameters within the UCS algorithm of our DEFT-UCS framework are selected manually using task specific knowledge. Future work should consider how to automate the selection of these hyper-parameters. Additionally, while our UCS algorithm within DEFT-UCS uses the distance between data samples and centroid distance to define sampling methods, future work should explore other sampling methods informative to NLP tasks. Additionally, we show the benefit of DEFT-UCS in the context of text generation across eight text-editing datasets. However, future work should benchmark DEFT-UCS across other diverse NLP tasks, beyond text generation, such as summarization or text expansion. Moreover, while our human evaluation shows that DEFT-UCS produced models generate accurately edited texts, future work should conduct larger-scale user studies to understand human perceptions of across multiple qualtitative dimensions. More work is also required to explore the benefit of DEFT-UCS in fine-tuning other PLMs for downstream NLP tasks, and comparing the benefits of DEFT-UCS with PEFT (Fu et al., 2023; Hu et al., 2021) approaches, and whether DEFT-UCS with PEFT can further improve the fine-tuning efficiency of PLMs.

Ethics Statement We utilize a publicly available dataset from CoEDIT[4]. The dataset primarily focuses on non-meaning changing text edits and does not raise any privacy concerns. Nevertheless, the underlying autoregressive models may hallucinate and propagate biases. Before deploying for real world applications, considerations on how to incorporate user feedback for continual system improvement should be studied. Additionally, we have acknowledged the limitations of our DEFT-UCS framework and the need for more extensive benchmarking with various other PLMs and downstream tasks. Our work provides a initial set of results and is an effort to motivate further research in data-efficient fine-tuning of PLMs.

[4]https://huggingface.co/datasets/grammarly/coedit

References

Fernando Alva-Manchego, Louis Martin, Antoine Bordes, Carolina Scarton, Benoît Sagot, and Lucia Specia. 2020. Asset: A dataset for tuning and evaluation of sentence simplification models with multiple rewriting transformations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4668–4679.

Jean-Michel Attendu and Jean-Philippe Corbeil. 2023. Nlu on data diets: Dynamic data subset selection for nlp classification tasks. *arXiv preprint arXiv:2306.03208*.

Vighnesh Birodkar, Hossein Mobahi, and Samy Bengio. 2019. Semantic redundancies in image-classification datasets: The 10% you don't need. *arXiv preprint arXiv:1901.11409*.

Tuhin Chakrabarty, Vishakh Padmakumar, and He He. 2022. Help me write a poem: Instruction tuning as a vehicle for collaborative poetry writing. *arXiv preprint arXiv:2210.13669*.

Mayee F Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. 2023. Skill-it! a data-driven skills framework for understanding and training language models. *arXiv preprint arXiv:2307.14430*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Shizhe Diao, Pengcheng Wang, Yong Lin, and Tong Zhang. 2023. Active prompting with chain-of-thought for large language models. *arXiv preprint arXiv:2302.12246*.

Wanyu Du, Vipul Raheja, Dhruv Kumar, Zae Myung Kim, Melissa Lopez, and Dongyeop Kang. 2022. Understanding iterative revision from human-written text. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3573–3590.

Jane Dwivedi-Yu, Timo Schick, Zhengbao Jiang, Maria Lomeli, Patrick Lewis, Gautier Izacard, Edouard Grave, Sebastian Riedel, and Fabio Petroni. 2022. Editeval: An instruction-based benchmark for text improvements. *arXiv preprint arXiv:2209.13331*.

Joseph L Fleiss and Jacob Cohen. 1973. The equivalence of weighted kappa and the intraclass correlation coefficient as measures of reliability. *Educational and psychological measurement*, 33(3):613–619.

Zihao Fu, Haoran Yang, Anthony Man-Cho So, Wai Lam, Lidong Bing, and Nigel Collier. 2023. On the effectiveness of parameter-efficient fine-tuning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12799–12807.

Sariel Har-Peled and Akash Kushal. 2005. Smaller coresets for k-median and k-means clustering. In *Proceedings of the twenty-first annual symposium on Computational geometry*, pages 126–134.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Hamish Ivison, Noah A Smith, Hannaneh Hajishirzi, and Pradeep Dasigi. 2022. Data-efficient finetuning using cross-task nearest neighbors. *arXiv preprint arXiv:2212.00196*.

Krishnateja Killamsetty, Xujiang Zhao, Feng Chen, and Rishabh Iyer. 2021. Retrieve: Coreset selection for efficient and robust semi-supervised learning. *Advances in Neural Information Processing Systems*, 34:14488–14501.

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

Shiye Lei and Dacheng Tao. 2023. A comprehensive survey to dataset distillation. *arXiv preprint arXiv:2301.05603*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

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. *arXiv preprint arXiv:2309.04564*.

Sewon Min, Mike Lewis, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. MetaICL: Learning to learn in context. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2791–2809, Seattle, United States. Association for Computational Linguistics.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. 2020. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. 2023. Orca: Progressive learning from complex explanation traces of gpt-4. *arXiv preprint arXiv:2306.02707*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. *arXiv preprint arXiv:1702.04066*.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B Hall, Daniel Cer, and Yinfei Yang. 2021. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. *arXiv preprint arXiv:2108.08877*.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34:20596–20607.

Reid Pryzant, Richard Diehl Martinez, Nathan Dass, Sadao Kurohashi, Dan Jurafsky, and Diyi Yang. 2020. Automatically neutralizing subjective bias in text. In *Proceedings of the aaai conference on artificial intelligence*, volume 34, pages 480–489.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Vipul Raheja, Dhruv Kumar, Ryan Koo, and Dongyeop Kang. 2023. Coedit: Text editing by task-specific instruction tuning. *arXiv preprint arXiv:2305.09857*.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Timo Schick, Jane Dwivedi-Yu, Zhengbao Jiang, Fabio Petroni, Patrick Lewis, Gautier Izacard, Qingfei You, Christoforos Nalmpantis, Edouard Grave, and Sebastian Riedel. 2022. Peer: A collaborative language model. *arXiv preprint arXiv:2208.11663*.

Ketan Rajshekhar Shahapure and Charles Nicholas. 2020. Cluster quality analysis using silhouette score. In *2020 IEEE 7th international conference on data science and advanced analytics (DSAA)*, pages 747–748. IEEE.

Lei Shu, Liangchen Luo, Jayakumar Hoskere, Yun Zhu, Canoee Liu, Simon Tong, Jindong Chen, and Lei Meng. 2023. Rewritelm: An instruction-tuned large language model for text rewriting. *arXiv preprint arXiv:2305.15685*.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536.

Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, et al. 2022. Selective annotation makes language models better few-shot learners. *arXiv preprint arXiv:2209.01975*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners. *ArXiv*, abs/2109.01652.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016a. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016b. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics*, 4:401–415.

Shuo Yang, Zeke Xie, Hanyu Peng, Min Xu, Mingming Sun, and Ping Li. 2022. Dataset pruning: Reducing training data by examining generalization influence. *arXiv preprint arXiv:2205.09329*.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. 2023. Metamath: Bootstrap your own mathematical questions for large language models. *arXiv preprint arXiv:2309.12284*.

Yue Zhang, Leyang Cui, Deng Cai, Xinting Huang, Tao Fang, and Wei Bi. 2023. Multi-task instruction tuning of llama for specific scenarios: A preliminary study on writing assistance. *arXiv preprint arXiv:2305.13225*.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. 2023a. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*.

Daquan Zhou, Kai Wang, Jianyang Gu, Xiangyu Peng, Dongze Lian, Yifan Zhang, Yang You, and Jiashi Feng. 2023b. Dataset quantization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 17205–17216.

A DEFT-UCS Applied to CoEDIT

A.1 CoEDIT Dataset Details

The CoEDIT dataset, D_{coEDIT}, from Raheja et al. (2023) is comprised of several edit tasks, including fluency, coherence, clarity, paraphrasing, neutralization and formalization. As mentioned in (Raheja et al., 2023), the 82k data samples follow the format of $\langle instruction : source, target \rangle$ pairs. The source and target pairs come from a variety of different datasets related to each editing task. Table 4 summarizes the datasets utilized to represent each edit task in D_{CoEDIT}. The *instruction* component are task-specific and generated from a pool of instructional prompts. For example, for a grammar correction task, an instruction could be "Fix grammar errors" or "Fix grammatical errors in this sentence". The list of all instructional prompts utilized are detailed in (Raheja et al., 2023).

Edit Task	Datasets in D_{coEDIT}
Fluency	NUCLE-14 Lang-8 BEA-19
Coherence	DiscoFuse
Clarity (Simplification)	NEWSELA WikiLarge WikiAuto ParabankV2 Iterator-Clarity
Paraphrasing	ParabankV2
Formalization	GYAFC
Neutralization	WNC

Table 4: Data in D_{CoEDIT}(Raheja et al., 2023) is comprised of samples from the above datasets. This table is a simplified version of Table 1 in Raheja et al. (2023).

A.2 DEFT-UCS Model Fine-Tuning Details

Recall that all DEFT-UCS models in this paper are produced by fine-tuning Flan-T5 Large (Chung et al., 2022). We fine-tune Flan-T5 Large such that we can make accurate comparisons with M_{CoEDIT}(Raheja et al., 2023) which represents a fine-tuned Flan-T5-Large model on D_{CoEDIT}. Furthermore, to remove any difference in model performances due to differing hyperparameters, we utilize the hyperparameters listed in Raheja et al. (2023). Specifically, we use the Adam optimizer with a learning rate of 1e-4. All DEFT-UCS models in the main paper are trained for 5 epochs with early stopping and the model checkpoints with the best validation loss are saved. To perform fine-tuning, we leverage 4 A10G GPUs, from AWS G5 instances, using Deepspeed (Rasley et al., 2020), and the maximum source and target sequence length is set to 256.

B Embedding Representations in UCS

B.1 Representation Details

For K-means clustering to learn informative clusters, selecting the right latent space representation for the input data is important. In our application, an accurate embedding representation should allow each cluster to predominantly represent a certain type of editing task. For example all data related to editing for fluency should be clustered together, whereas all data related to grammar correction should be clustered together. To ultimately select an accurate embedding representation, we experimented with three different representations: sentence-level encoding from Sentence-T5 (Ni et al., 2021), BART CLS token embedding, as well as an averaged word token embedding from Flan-T5. As a brief summary, Sentence-T5 (Ni et al., 2021) maps sentences to a 768 dimensional vector space using only the encoder from T5. Specifically, Ni et al. (2021) demonstrate that Sentence-T5 embeddings are able to lead to high performance in sentence transfer tasks. Similarly, we also experiment with BART (Lewis et al., 2019) CLS token embeddings, inspired by the notion that CLS token can provide informative representations of the input sentence for downstream tasks (Devlin et al., 2018). We also experiment with an average pooling method of averaging all word embeddings of an input sequence, using the Flan-T5 model, to reach a sentence-level embedding.

B.2 Representation Analysis

Figure 5 demonstrates the K-means clustering results for each sentence-level embedding representation. Overall, we find that Sentence-T5 provides the strongest sentence-level embedding that allows the clustering algorithm to best separate input data based on its related editing task. Specifically, when analyzing Figure 5(a), we see that each cluster is largely comprised of a single edit-task. For example, cluster 1 largely includes data related to "paraphrasing", while cluster 4 largely includes data related to improving "coherence". In Figure 5(b) and Figure 5(c) we observe that the task specific data is more distributed among several clusters, indicating weaker cluster separation among the different editing task related data. Although the clusters formed via Sentence-T5 embeddings (Ni et al., 2021) are not perfect, they offer the strongest separation of task-related data compared to the other embedding representations. Given these results, we leverage Sentence-T5 as our latent space representation when performing UCS.

C Evaluation Dataset Details

For all datasets used in our evaluation, we utilize the publicly available test splits from each dataset. To each data sample (source and target pair), we prepend a randomly selected instructional prompt related to the edit task. For example, for all test samples from TurkCorpus, we prepend a randomly selected instructional prompt from the text simplification choices provided in Raheja et al. (2023). In Table 5 we provide example test data samples from each evaluation dataset. For context, we additionally provide the sizes of the test splits available for each evaluation dataset. The test splits are as follows: TurkCorpus includes 359 test data samples, Asset includes 359, Iterator Coherence includes 36, Iterator Clarity contains 186, Iterator Fluency contains 88, JFLEG contains 748 and WNC contains 1000. Note, we additionally evaluate on a combined Iterator dataset, noted as Iterator Global in Table 1, which includes all test samples from Iterator Coherence, Clarity and Fluency. The motivation of including an Iterator Global evaluation dataset is to understand model performances on a more generic style-editing task (Du et al., 2022). Furthermore, in Figure 6, we provide a TSNE visualization of the evaluation datsets, particularly embedding representations of all source sentences using Sentence-T5 (Ni et al., 2021). The visualization demonstrates the diversity among the different datasets, and highlight that the evaluation tasks are not all semantically similar.

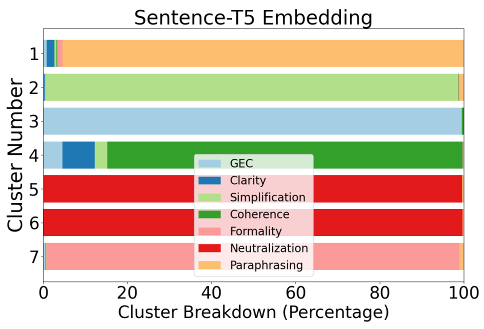
D Additional DEFT-UCS Results

D.1 Extended Best DEFT-UCS Analysis

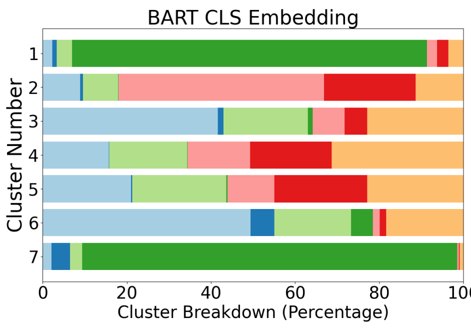
In Section 7.3, we demonstrate that utilizing on 32.5% of D_{CoEDIT} can result in an overall best DEFT-UCS model that surpasses M_{CoEDIT} (Raheja et al., 2023) SARI and ROUGE-L scores on 6 of the 8 evaluation datasets. While Figure 3 in the main paper provides an analysis using up to 45% of D_{CoEDIT}, in this section, we include Figure 7 which provides an exhaustive analysis using up to 87.5% of D_{CoEDIT}. From Figure 7, we observe that to surpass SARI and ROUGE-L scores on 7 out of the 8 evaluation datasets, 47.5% of D_{CoEDIT} is necessary. Additionally, we observe that while 75% of D_{CoEDIT} can be leveraged to surpass ROUGE-L scores on all evaluation datasets. Overall, these results indicate a trade-off between marginal improvement in model performance and the amount of additional data required.

D.2 Additional Qualitative Analysis

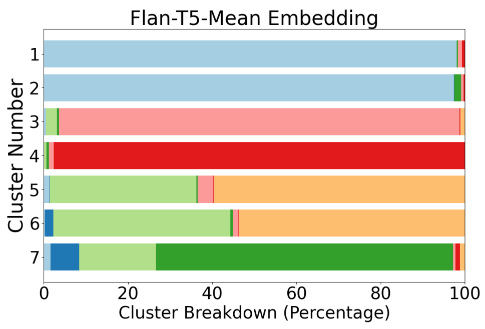
In Table 6, we present example model outputs, qualitatively comparing M_{CoEDIT}, $M^{Flan-T5-LG}_{DEFT-UCS}$, $M_{LLAMA-7B}$ and $M_{BLOOM-560M}$. Overall, we observe that the example sentences generated by $M^{Flan-T5-LG}_{DEFT-UCS}$ and M_{CoEDIT} are either identical or similarly edit the input sentence to reflect the edit instruction. When we examine the zero-shot inference outputs from $M_{LLAMA-7B}$ and $M_{BLOOM-560M}$ we observe that these models are not able to produce accurately edited sentences. Instead, we notice repeated generation from both $M_{LLAMA-7B}$ and $M_{BLOOM-560M}$ as well as additional generations that are tangential. These repeated, longer, and irrelevant generated sentences also explain the much lower ROUGE-L observed in Table 2 within the main paper. Overall, these generated outputs from each model provide further understanding of the need for instruction-tuned LLMs for tasks such as text-editing. These generated output examples also re-iterate that our DEFT-UCS model, $M^{Flan-T5-LG}_{DEFT-UCS}$, can generate similarly edited sentences to the CoEDIT baseline, M_{CoEDIT}, while being fine-tuned on 70% less data.



(a)



(b)



(c)

Figure 5: Comparing the distribution of task-related data among clusters after performing K-Means when utilizing Sentence-T5 embedding (a), BART CLS embeddings (b) and averaged Flan-T5 word embeddings (c) for sentence representations.

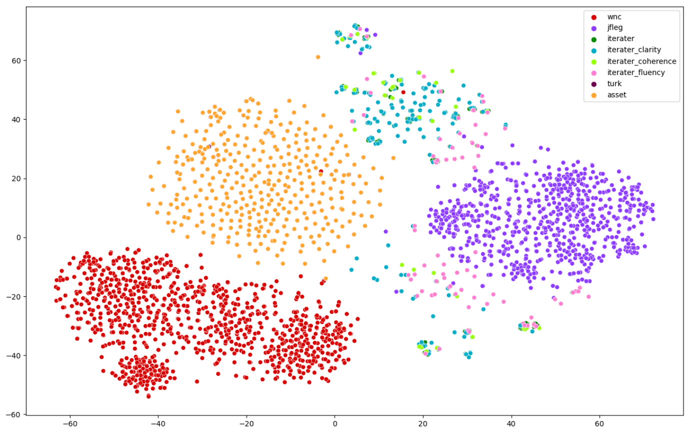
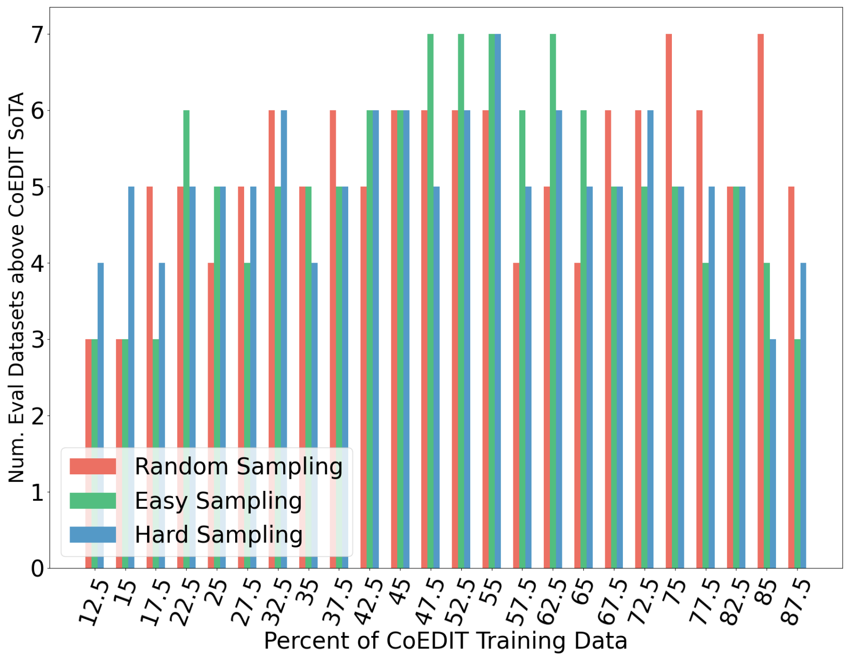


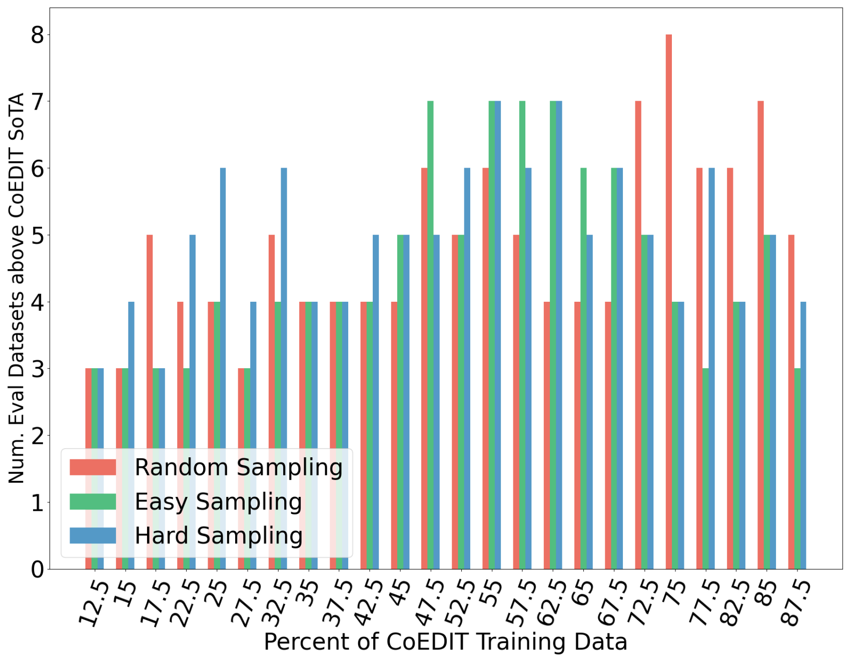
Figure 6: TSNE visualization of the source sentences within all evaluation datasets.

Evaluation Dataset	**Edit Task**	**Input Example**	**Output Example**
TurkCorpus (Xu et al., 2016a)	Text Simplification	*Make the sentence simple:* The great dark spot is thought to represent a hole in the methane cloud deck of neptune.	The great dark spot is thought to represent a hole in the methane.
Asset (Alva-Manchego et al., 2020)	Simplification	*Simplify this sentence:* She remained in the United States until 1927 when she and her husband returned to France.	She remained in the United States until returning to France with her husband in 1927.
Iterator Coherence (Du et al., 2022)	Coherence	*Fix sentence flow:* Based on the general linguistic structure of humor, in this paper, we propose a novel approach for detecting humor in short texts by using BERT sentence embedding.	In this paper, we propose a novel approach for detecting humor in short texts by using BERT sentence embedding .
Iterator Clarity (Du et al., 2022)	Clarity	*Write a clearer version for the sentence:* Using our human-evaluation datasets, we show that existing metrics based on n-gram similarity do not correlate with human judgments.	Using our human-evaluation datasets, we show that widely used n-gram similarity do not correlate with human judgments.
Iterator Fluency (Du et al., 2022)	Fluency	*Fix disfluencies in the sentence:* In addition, we provide the first robust corpus this kind for the Brazilian Portuguese language.	In addition, we provide the first robust corpus of this kind for the Brazilian Portuguese language.
JFLEG (Napoles et al., 2017)	Grammar Correction	*Fix the grammar mistakes:* Every person needs to know a bit about math, sciences, arts, literature and history in order to stand out in society.	Every person needs to know a bit about math, science, art, literature and history in order to stand out in society.
WNC (Pryzant et al., 2020)	Neutralization	*Remove points of view:* During the unnecessary horseplay, Hamlin fell and severely injured his hand.	During the horseplay, Hamlin fell and severely injured his hand.

Table 5: Example data samples for each evaluation dataset. Note, the instructional prompt (italicized) are randomly sampled from a list of instructional prompts available in Raheja et al. (2023).



(a)



(b)

Figure 7: Presenting a full analysis, utilizing up to 87.5% of D_{CoEDIT}, on the different sampling methods and amounts of data needed to beat soTA M_{CoEDIT} performance, considering SARI (a) and ROUGE-L (b) scores.

	Generated Model Outputs			
Input	M_{CoEDIT}	$M^{Flan-T5-LG}_{DEFT-UCS}$	$M_{LLaMA2-7B}$	$M_{BLOOM-560M}$
Make this easier to understand: the term "protein" itself was coined by berzelius, after mulder observed that all proteins seemed to have the same empirical formula and might be composed of a single type of -lrb- very large -rrb- molecule.	The term "protein" itself was created by berzelius in 1879, after mulder observed that all proteins seemed to have the same empirical formula and might be composed of a single type of very large molecule.	berzelius thought that all proteins seemed to have the same empirical formula and might be composed of a single type of molecule.	It is a protein made by bacteria that is used to make a large number of other proteins. This is a very large protein that is used to make a large number of other proteins. This is a very large protein that is used to make a large number of other proteins. It is a protein made by bacteria that is used to make a large number of other proteins. It is a protein made by bacteria that is used to make a large number of other proteins.	The term was coined by mulder to describe the fact that the protein was composed of a single type of -lrb- molecule. The term was coined by mulder to describe the fact that the protein was composed of a single type of -lrb- molecule. The term was coined by mulder to describe the fact that the protein was composed of a single type of -lrb- molecule.
Remove non-neutral POVs: jimi hendrix (musician), great musician and vocalist died while at girlfriend's monika dannemann hotel room in london.	jimi hendrix (musician), musician and vocalist died while at girlfriend's monika dannemann hotel room in london.	jimi hendrix (musician), musician and vocalist died while at girlfriend's monika dannemann hotel room in london.	This was an interesting read. I think I'm going to have to edit that article. I've been doing some research on this, and it seems that the story is that Hendrix was at the hotel with Monika Dannemann. She was a German woman who had been a friend of his for some time. She was a model and an artist, and she had been a fan of his music since she was a teenager. She had been in London with him for a couple of weeks, and she had been helping him out with his music.	the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week. the singer was in a coma for a week.
Fix grammaticality in this sentence: The important thing about this case is , each countrey hase to use official solution for their learning problems.	The important thing about this case is that each country has to use an official solution for their learning problems.	The important thing about this case is that each country has to use an official solution for their learning problems.	The important thing about this case is that each country has to use its own official solution for their learning problems. The important thing about this case is that each country has to use its own official solution for their learning problems.	The main purpose of this paper is to study the grammaticality of the sentence: The important thing about this case is , each countrey hase to use official solution for their learning problems. The main purpose of this paper is to study the grammaticality of the sentence: The important thing about this case is, each countrey hase to use official solution for their learning problems.
Fix disfluencies in the sentence: To work hard means to run the company.	To work hard means to run a company.	To work hard means to run a company.	Disfluencies are speech disruptions that are often caused by an inability to find the right word. Disfluencies are common in everyday speech. Examples of disfluencies in English 1. To work hard means to run the company. 2. To work hard means to run the company	The sentence is not a sentence, but a sentence with a disfluency. The sentence is not a sentence, but a sentence with a disfluency. The sentence is not a sentence, but a sentence with a disfluency. The sentence is not a sentence, but a sentence with a disfluency. The sentence is not a sentence, but a sentence with a disfluency.

Table 6: Example generated, edited sentences from each model for a given input. We observe that non-instruction tuned LMs such as BLOOM-560M and LLAMA-7B mostly struggle in zero-shot inference as demonstrated by the repeated or irrelevant generation. Raheja et al. (2023)20312