Sample Ordering and Selection Both Matter: A Case Study on the Impact of Sample Ordering in Active Learning for Translation

Anonymous ACL submission

Abstract

Active learning (AL) is a technique for efficiently selecting subsets of data for annotation and fine-tuning, which has been shown to outperform random sampling in classification tasks. However, it remains unclear why applying similar strategies does not consistently lead to similar gains in performance on natural language generation tasks. We hypothesize that previous methods underperform random sampling as they rarely consider interactions between the selected samples, and thus overlook training dynamics which may impact model performance. We find that in machine translation (MT), the ordering of the samples has a significant impact on performance, and show that fine-tuning the model on multiple shuffles of the data can allow AL to outperform random sampling in cases where it previously did not. We then present ways in which some shuffles of the training data learn the task of MT suboptimally, to motivate future AL strategies to explicitly account for training dynamics and mitigate these failure modes.

1 Introduction

Active learning (AL) is a training paradigm used to efficiently select unlabeled data to annotate which yields the best model performance. This is useful when annotation resources are constrained, like for machine translation (MT) in low-resource languages. AL often works by greedily selecting the most informative data at each iteration, based on various metrics of model uncertainty or data diversity and representativeness. This has successfully been applied to various image and text classification tasks (Kirsch et al., 2019; LaBonte et al., 2022; Gal and Ghahramani, 2016; Zhang et al., 2017; Ein-Dor et al., 2020; Prabhu et al., 2019). However, it remains unclear why applying AL strategies in text generation does not achieve similar gains in performance (Perlitz et al., 2023). This is surprising, as it suggests that models do not reliably benefit from being trained on samples deemed as informative according to AL metrics.

We aim to understand why AL fails to outperform random sampling in text generation. One hypothesis is that the poor performance stems from the fact that current methods only focus on maximizing informativeness metrics at each round, and fail to account for interactions between the samples selected across rounds. Hence, there may be unnoticed training dynamics, which make the complex patterns and representations in generation tasks even harder to learn. In this work, we probe whether training dynamics may be impacting the performance of AL strategies. In particular, we ask: **To what extent do training dynamics explain the variance in model performance, compared to informativeness metrics which AL optimize for?**

Overall, we find that the ordering of the samples significantly affects the model's performance. We demonstrate that models can achieve better performance by using AL *and* accounting for ordering, by rerunning the model with different shuffles of the data. In some cases, fine-tuning on data chosen by AL with the optimal ordering outperforms data selected randomly, when it previously did not. Hence, future AL work should focus not only on what samples to select, but also on how to account for ordering and dynamics between the selected data. Our analysis is structured in three parts.

We first validate the underperformance of AL in MT, by showing that using AL strategies to choose data to fine-tune MT models on does not yield better performance than randomly choosing (Figure 1). Even an oracle, where we use the gold-label translations to choose samples which the model performs poorly on, does not outperform random sampling (Figure 1).

We then show that the ordering of the samples have a considerable impact on model performance. Specifically, we first find that metrics of information content which AL strategies optimize for are

only weakly correlated with performance. We then find that the variance in performance is explained more by the ordering of samples than the choice of samples themselves. We show that by finding "better" orderings, models achieve better performance using data selected with AL strategies. In some cases, taking the best model across multiple shuffles allows AL to outperform random sampling.

Finally, we perform a case study to understand the ways in which certain orderings of the data negatively impact the model's performance, to motivate future work to address these failure modes. We analyze an English-Filipino MT dataset, as Filipino is one of the languages which has abundant unlabeled data, but scarce labeled data (Joshi et al., 2020), making it an ideal candidate for AL. We find that in some shuffles of the data, the model learns incorrect patterns or distorted representations, in ways that it does not recover from. While it is unclear what characteristics of an ordering of data results in suboptimal performance, our findings underscore the need to avoid these suboptimal runs by finding the "better" orderings of the training data. Hence, future work in AL should consider ways to optimize both for the data informativeness and training dynamics, both in MT and possibly in other generation tasks.

2 Related Work

Active Learning Active learning (AL) is a training paradigm where data is iteratively selected, annotated, and added to the training pool from a set of unlabeled candidates (Cohn et al., 1996). AL has been used to efficiently select subsets that achieve better performance than random sampling on image (Kirsch et al., 2019; LaBonte et al., 2022; Gal and Ghahramani, 2016) and text classification (Zhang et al., 2017; Ein-Dor et al., 2020; Prabhu et al., 2019; Siddhant and Lipton, 2018) tasks. However, Perlitz et al. (2023) found that AL strategies did not outperform a random baseline for generation tasks when choosing 100-500 samples. This may hinder the use of AL in machine translation (MT) for low-resource settings, where reducing annotation costs would be most beneficial, as specialized annotation can cost up to $5 USD/sentence (Labs, 2025). We analyze the systematic underperformance, to better understand AL in the very low-resource setting.

Active Learning in Machine Translation In this work, we focus on machine translation (MT). In contrast to most work applying AL to MT which uses thousands of examples (Zhao et al., 2020; Zeng et al., 2019; Mohiuddin et al., 2022; Chimoto and Bassett, 2022), we constrain the number of samples to 100 as Perlitz et al. (2023) did, to reflect a very low-resource setting scenario. While we focus on MT, our insights may extend to AL in other text generation tasks.

Acquisition Functions Work in AL often focuses on the acquisition function – the strategy for selecting samples. According to Zhang et al. (2022) there are two broad categories: **Representativeness** strategies maximize the diversity of the training examples selected, measured using word-based (Zhao et al., 2020; Zeng et al., 2019) or embedding-based (Sener and Savarese, 2018) metrics. **Uncertainty** strategies choose samples which the model is most uncertain about and, thus, from which the model is assumed to learn the most information. These use token probability or entropy (Zhao et al., 2020; Mohiuddin et al., 2022), variance in model responses (Gal et al., 2017; Schmidt et al., 2022; Liu and Yu, 2023; Zeng et al., 2019), or predicted quality scores (Chimoto and Bassett, 2022). In our work, we validate the effectiveness of these strategies in MT and analyze the relationship between these metrics and model performance.

3 Validation Study

We try AL strategies in MT, to validate that AL fails to consistently outperform random sampling.

AL Algorithm At each iteration, we choose a subset $\mathcal{S}_i$ from an unlabeled dataset $\mathcal{D}$ using acquisition function f_{aq}, label it, and fine-tune a model θ on it, with the goal of maximizing performance on a test set at each iteration (Algorithm 1).

Algorithm 1 Active Learning Framework

Require:
$\mathcal{D}$ (Unlabeled Dataset)
b (Budget per Round), n (Num Rounds)
θ (Language Model)
f_{aq} (Acquisition Function)
for $i \leftarrow 1$ **to** n **do**
  for $j \leftarrow 1$ to $|\mathcal{D}|$ **do**
    $\text{score}_j \leftarrow f_{\text{aq}}(\mathcal{D}_j, \theta)$
  end for
  $\mathcal{S}_i \leftarrow \text{argmax}_{I \subset \{1,\cdots,n\}:|I|=b} \sum_{i \in I} \text{score}_i$
  Finetune θ on $\mathcal{S}_i$
  $\mathcal{D} \leftarrow \mathcal{D} \backslash \mathcal{S}_i$
end for

Implementation Details For the acquisition function f_{aq}, we use average token probability and entropy (Zhao et al., 2020), lexical similarity (Schmidt et al., 2022), BALD (Gal and Ghahramani, 2016), Greedy Core Set (Sener and Savarese, 2018), Delfy (Zhao et al., 2020) (See Appendix A).

We test MBART 50 (Tang et al., 2020), a pretrained multilingual model that can be fine-tuned on one GPU, making it relatively more accessible to fine-tune than LLMs. We use 10K samples from NLLB (Team et al., 2022) as the unlabeled dataset $\mathcal{D}$, and select $b = 100$ samples at each round with the largest score from f_{aq}. We test the model on FLORES-Plus (NLLB Team et al., 2024).

Results We find that selecting new training samples using AL strategies does not consistently outperform selecting them by randomly sampling (See Figure 1, left). Note that we define *outperforming* as achieving higher performance across all rounds, as we want a strategy that beats random for any n.

Seeing as the AL strategies do not beat random, we explore the performance of an oracle: at each iteration, we identify the samples which the model performs *worst* on using the gold-label translations (pick-worst), measured by the individual ChrF score for the sample against the reference. We *assume* the model will improve *more* when trained on samples it performs poorly on. For comparison, we also try the opposite strategy, choosing samples which the model performs well on (pick-best).

Like Perlitz et al. (2023), we find that the pick-worst strategy does not consistently achieve higher test set performance than random sampling across all the rounds of AL (See Figure 1, right). Given that various AL strategies do not outperform random sampling, we must re-evaluate the assumption that choosing data that maximizes informativeness metrics yields better performance.

4 Analysis Setup

Given that AL is unable to consistently outperform random sampling, we investigate the assumption of AL that maximizing the informativeness of a dataset, as measured by metrics of uncertainty or representativeness, leads to better performance. For this analysis, we adopt the following set up:

Model We use Multilingual BART 50 Base (MBART) (Tang et al., 2020); MBART was pretrained on 50 languages; we use a multilingual model to reflect a real-world setting where users apply AL when fine-tuning a pretrained model, which has seen multilingual data. Unless stated, we use a batch size of 8, learning rate of 5e-5, and 5 epochs.

Datasets We use four language pairs from NLLB (Team et al., 2022): English-Afrikaans (Eng-Afr), English-German (Eng-Deu), English-Filipino (Eng-Fil), and English-Hatian Creole (Eng-Hat) for fine-tuning; we sample 10K sentence pairs for the candidate set. We use FLORES Plus (NLLB Team et al., 2024) as our test set. We select these languages to test the model's behavior when the language is in the pre-training data (Afrikaans, German, and Filipino) and when it is not (Hatian Creole).

Evaluation In all analyses, we use the average ChrF+ score (Popović, 2017), which is a character-level F1 score shown to correlate well with human ratings in translation tasks, over the test set.

5 Results

5.1 To what extent do training dynamics explain the variance in performance, compared to metrics of informativeness which AL typically optimize for?

Selecting different subsets yields varying performance We verify if using different subsets of the data yields different levels of performance. We sample 500 subsets with 100 samples each from the candidate set. For each subset, we finetune MBART and evaluate on the test set, and plot the distribution of resulting test set ChrF+ scores.

Figure 2 shows a wide variance in performance across subsets, confirming that the choice of subset impacts the performance. We analyze if the data's information content, measured by AL metrics, explain the performance across subsets.

Performance is only weakly associated with metrics of information content optimized for by AL To understand the relationship between AL metrics and performance, we measure the correlation between ChrF scores and various AL metrics. For **representativeness** metrics, we compute (1) DelFy (Zhao et al., 2020) - a word frequency metric with a penalty for previously seen words, (2) L2 Distance (Ni et al., 2022; Sener and Savarese, 2018) - the average L2 distance of training examples from the center[1], and (3) word-level statistics of the dataset: the vocabulary size of the train set, the percentage

[1]Computed with the hidden state of the encoder's last layer; Center is the average embedding over the training examples

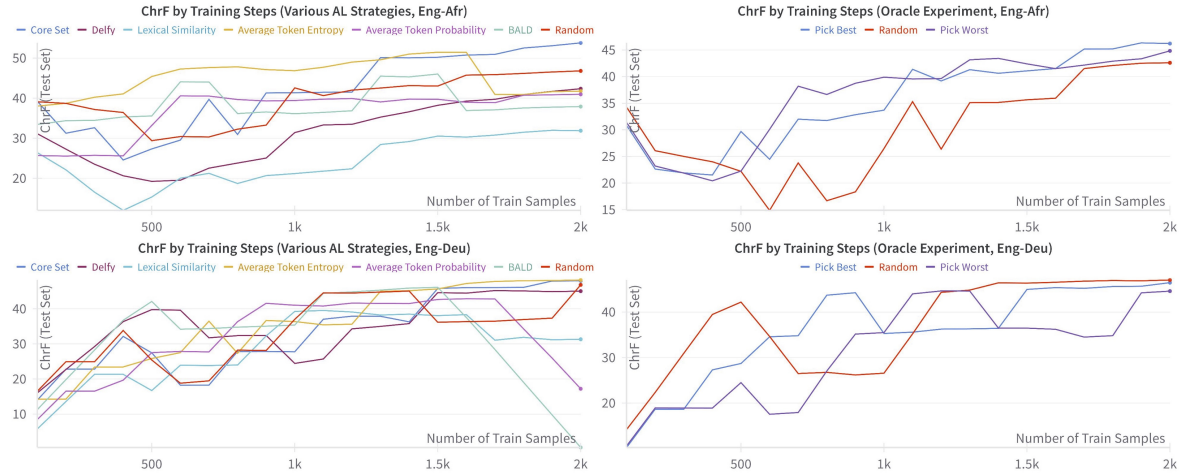

Figure 1: Various AL strategies are unable to consistently achieve better test set performance than random sampling; Plot shows test set performance (ChrF+) per AL round, plotted with smoothing over 5 steps

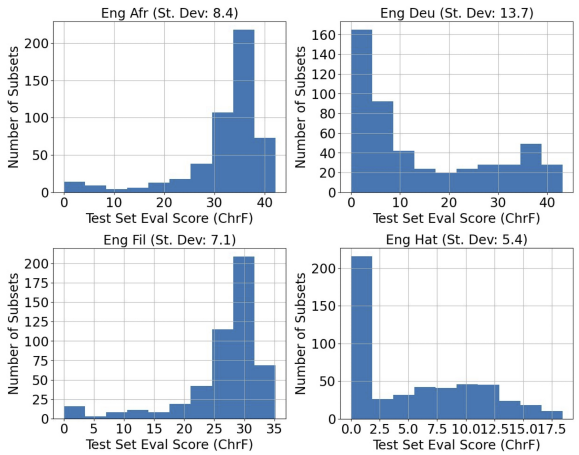

Figure 2: Fine-tuning on different subsets of the data yields considerable variance in test set performance; Plotted using 500 subsets with 100 samples each

		Afr	Deu	Fil	Hat
Representative	Number of Vocab	0.16	-0.15	0.08	-0.05
	% Shared Vocab	0.19	-0.06	0.07	-0.10
	% Test Vocab Seen	0.19	-0.08	0.07	-0.10
	Delfy (Source)	-0.09	0.02	0.04	0.03
	Delfy (Target)	-0.04	-0.02	0.01	0.10
	L2 Distance	0.03	-0.21	0.04	-0.04
Uncertainty	Mean Token Prob (Min)	-0.03	-0.07	-0.02	0.02
	Mean Token Prob (Avg)	-0.09	-0.08	-0.05	0.07
	BALD	0.07	-0.05	0.02	0.01
	Lexical Similarity	0.12	-0.07	0.08	-0.06
	Mean Token Entropy	-0.09	0.05	-0.14	0.04

Table 1: Spearman correlation between AL metrics and model performance (Test Set ChrF)

of vocabulary shared by train and test sets, and the percentage of test set vocabulary present in the train set (Appendix A.1).

For **uncertainty** metrics, we compute metrics by sample, then average over the dataset. We compute average token probability and entropy (Zhao et al., 2020), lexical similarity (Schmidt et al., 2022), and BALD score (Gal et al., 2017) (Appendix A.2).

As shown in Table 1, informativeness metrics are only weakly correlated with performance. Representativeness metrics are more strongly correlated with performance, but only achieve 19% at most.

To understand how much of the variance in performance these metrics explain, we regress the ChrF+ scores on the features computed above using ordinary least squares, and find that these features jointly explain only 4.1% (Eng-Afr), 5.1% (Eng-Deu), 2.0% (Eng-Fil), and 2.5% (Eng-Hat) of the total variance in performance (using R^2). This suggests that metrics of informativeness which AL optimizes for only loosely determine performance. This challenges the underlying assumption that optimizing for such metrics yields better performance.

In fact, most of the variance in performance can be attributed to the ordering of the samples of the data, rather than the samples themselves Given that the metrics of informativeness do not explain the variance in performance, we turn our attention to other sources of the observed variation. So far, the only variable we changed is the sample used to fine-tune the model. Thus, the only parameters we can change are the sampled data and the order in which they are shown to the model. We can decompose the variance in performance into the variance attributed to the samples, vs the ordering of those samples as follows, where G is the set of sampled subsets, each with N shuffles of the

same data, $p_{i,j}$ is the performance (ChrF+ score) from the i-th subset with the j-th ordering, $\bar{p}_i$ is the average performance for group i, and $\bar{p}$ is the average performance across all samples.

$$\underbrace{\frac{1}{NG}\sum_{i\in G}\sum_{j=1}^{N}(p_{i,j}-\bar{p})^2}_{\text{Total Variance}} = \underbrace{\frac{1}{NG}\sum_{i\in G}\sum_{j=1}^{N}(p_{i,j}-\bar{p}_i)^2}_{\substack{\text{Variance within Groups}\\ \text{(from Ordering)}}} + \underbrace{\frac{1}{G}\sum_{i\in G}(\bar{p}_i-\bar{p})^2}_{\substack{\text{Variance between Groups}\\ \text{(from Sampling)}}}$$

We study how much of the variance in performance is a result of the contents of the sample compared to the order of its elements. To do this, we fine-tune models on multiple shuffles of the same sample, and repeat this for multiple samples.

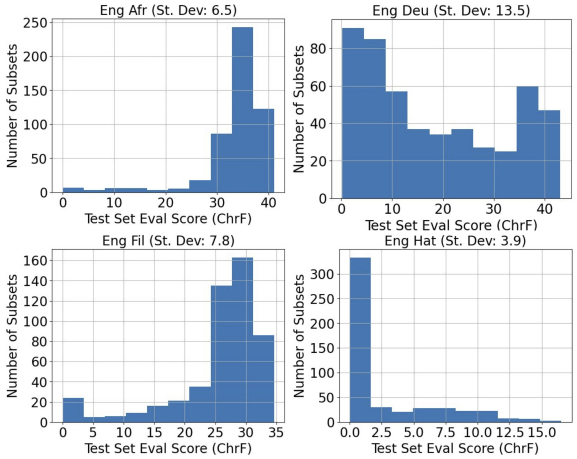

Figure 3: Reshuffling the same data yields considerable variance in performance; Each plot uses 500 shuffles of a set of 100 samples

We observe a large variance in performance from shuffling the data (Figure 3). In fact, the variance from shuffling the same data nears/exceeds the variance from sampling new data when comparing Figure 2 to 3 (Sample vs Shuffle: Eng-Afr: 8.4 vs. 6.5; Eng-Deu: 13.7 vs. 13.5; Eng-Fil: 7.1 vs. 7.8; Eng-Hat: 5.4 vs. 3.9). We repeat this with 2 to 200 samples with 20 re-orderings each. Using the above equation, we find that ordering explains much of the overall variance in performance (Figure 4). We repeat this using different batch sizes (8, 16, 32) and learning rates (1e-5, 5e-5, 1e-4), and still find that a majority of the variance is explained by ordering (Figure 5).

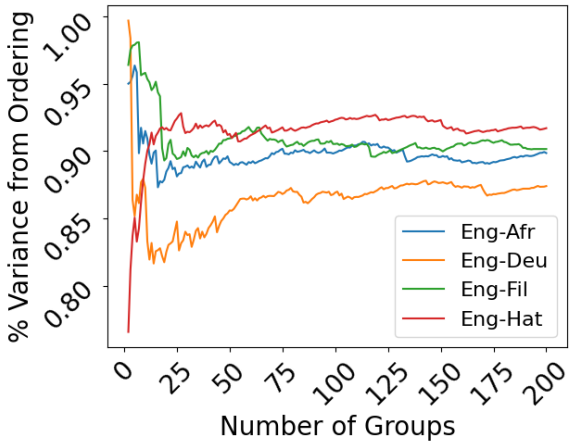

Figure 4: Across multiple groups of samples, each shuffled 20 times, the ordering of the samples accounts for between 80% to 95% of variance

5.2 How can these findings be used to improve AL performance?

So far, we demonstrated that the ordering of the samples have a large impact on performance, which suggests that these should be considered when using AL. We show that considering *both* ordering and AL can lead to better performance.

We take the data selected by various AL strategies, train the model on various shuffles of the same data, and take the model with the best result. As shown in Table 2, we are almost always able to find a shuffle of the data that leads to better performance. In some cases, shuffling results in an AL strategy outperforming random sampling, when it previously did not (highlighted in green).

It should be noted that this in itself is not a method, as we have optimized directly for test set performance in order to demonstrate that models *can* achieve better performance when trying different orderings. This aims to motivate future work in AL to find both the optimal subset *and* their optimal ordering simultaneously.

6 Case Study on Failure Modes Induced by Sample Ordering

The previous findings suggest that the informativeness of the samples (according to AL metrics) is only weakly associated with performance, whereas we expected this to impact performance most. In contrast, sample ordering has a much larger impact.

We study the behavior of models trained on different orderings of the same samples, looking at both the lexical and embedding level. Our aim is to

	Eng-Afr						Eng-Deu					
	500		1000		2000		500		1000		2000	
Strategy	NS	WS	NS	WS	NS	WS	NS	WS	NS	WS	NS	WS
BALD	**42.6**	**46.5**	1.8	**48.1**	47.0	**49.5**	**42.0**	**44.2**	43.9	**45.7**	0.1	**48.1**
Core Set	**47.7**	**50.6**	3.1	**52.0**	**54.4**	**54.4**	0.0	**47.0**	**47.7**	**47.7**	**47.8**	**49.2**
DelFy	26.7	29.6	31.6	39.1	43.2	43.2	**38.8**	41.2	42.0	43.8	**44.7**	46.6
Lex Sim	18.3	27.0	0.0	29.3	34.8	34.8	**39.4**	39.4	39.9	39.9	39.7	42.0
MTE	**48.4**	**49.9**	**49.4**	**51.2**	**52.1**	**53.2**	**45.5**	**45.8**	0.0	**46.9**	**47.7**	**48.6**
MTP	**41.7**	**44.3**	37.2	44.8	42.4	44.2	0.0	43.2	43.1	43.9	0.0	44.8
Random	34.2	41.5	43.1	46.0	47.3	49.1	0.0	43.9	44.0	45.4	47.4	47.4

	Eng-Fil						Eng-Hat					
	500		1000		2000		500		1000		2000	
Strategy	NS	WS	NS	WS	NS	WS	NS	WS	NS	WS	NS	WS
BALD	**40.2**	**40.2**	**38.2**	**40.8**	40.0	41.0	9.4	21.5	**24.4**	**29.3**	**29.9**	**32.6**
Core Set	**39.6**	**45.4**	**47.2**	**47.3**	**46.6**	**50.1**	2.9	**33.4**	0.9	**36.2**	**37.5**	**37.5**
DelFy	23.5	26.2	33.3	33.3	35.1	38.9	9.2	17.0	10.7	18.1	5.8	24.5
Lex Sim	17.1	21.0	31.2	31.2	33.6	33.6	**11.2**	11.9	3.9	11.3	15.2	15.2
MTE	**40.8**	**45.1**	**46.3**	**46.5**	**47.1**	**49.1**	0.6	**30.5**	**32.4**	**34.7**	11.6	**36.9**
MTP	**31.9**	**40.4**	**37.8**	39.6	40.1	40.6	0.6	**25.3**	**19.0**	**26.7**	17.0	27.4
Random	31.5	37.7	36.4	40.1	42.2	44.2	10.0	22.0	15.6	25.2	27.6	29.8

Table 2: Best test set ChrF+ score across 50 shuffles of the data selected using AL strategies; Shuffling the data post-AL yields improvements in nearly all strategies when using 500, 1000, and 2000 samples, sometimes outperforming random when the strategy initially did not; NS: No Shuffling (1st AL run), WS: With Shuffling; Runs that beat random sampling are written in **bold**, and those that beat it only *after* shuffling are highlighted in green

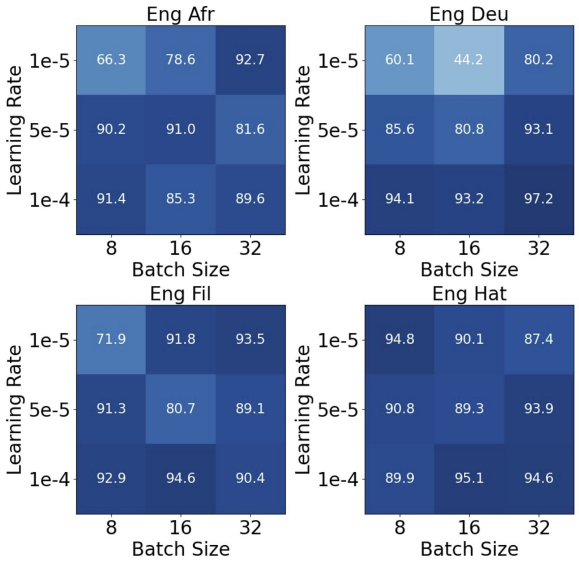

Figure 5: Percent of variance in test set performance measured by ChrF+ from ordering, computed using 20 samples shuffled 20 times each, on 25% of the original test set; Ordering accounts for most of the variance in performance across different hyperparameters

understand the ways in which some orderings learn MT suboptimally, in ways that others do not. By understanding the failure modes caused by certain orderings, we can better motivate future methods to understand the mechanism by which samples interact with each other, and use these to propose AL strategies that account for the sample ordering.

6.0.1 Some runs lead to poor acquisition of the task and unexpected interactions with the model's parametric knowledge

We fine-tune models on multiple shuffles of an English-Filipino task. We use a batch size of one to isolate the effect of each training sample. At each training step, we analyze how the predictions for the test set change. We observe the following:

In some orderings, the model learns incorrect translations of the vocabulary In one shuffle for example, at fine-tuning step 91, the model is trained on the word *panalangin*, which means prayer. After one or more fine-tuning steps, the model starts to incorrectly use that word in various test samples. In fact, at the end of fine-tuning, the model incorrectly generates the word *panalangin* in 253 out of 1012 test set examples (See Table 3). This suggests that the model generates the vocabulary en masse without necessarily learning its meaning. In contrast, in another shuffle of the data, the model is fine-tuned on the word *panalangin* at step 14, and does not exhibit this incorrect usage of the word.

In some orderings, the models learn less of the vocabulary words in the training data We see that models are unable to correctly learn certain Filipino vocabulary despite having been trained on them (Figure 6). Moreover, this failure to learn vocabulary is more severe in some shuffles of the data than others. In 4/5 shuffles of the same data,

Set	Step	Input	Target
Train	91	A **prayer** for our beloved nation.	Isang **panalangin** sa aming mahal na nasyon.
Set	**Step**	**Input**	**Prediction**
Test	94	The tenth named storm of the Atlantic Hurricane season, Subtropical Storm Jerry, formed in the Atlantic Ocean today.	Ang lalake ang huling named na na na named na ang **panalangin**..., ang **panalangin** sa Atlantic ng Amerika...
	95	The number of people present was so large that it was not possible for everybody to gain access to the funeral in St. Peter's Square.	Ang mga tao na ito ay hindi posible para sa lahat ng tao ang **panalangin** sa St. Peter's Square.
		Prime Minister Stephen Harper has agreed to send the government's 'Clean Air Act'... for review, before its second reading, after Tuesday's 25 minute meeting with NDP leader Jack Layton at the PMO.	Stephen Harper ay nag-iisa ang **panalangin** sa ang lahat ng mga tao para sa **panalangin**...
	97	The final match of the series will take place at Ellis Park in Johannesburg next week, when the Springboks play Australia.	Ang palangin sa Ellis Park sa Johannesburg, ang **panalangin** sa Australia ng mga tao ng mga tao ng Australia.

Table 3: Models incorrectly generate the word *panalangin* across various samples after being fine-tuned on an example with the word (**Red** indicates wrong usage of the word)

the model fails to generate at least one vocabulary word seen in the training data for 72.1% of test set examples. However in another shuffle, 85.1% of test samples have at least one Filipino word which the model does not generate.

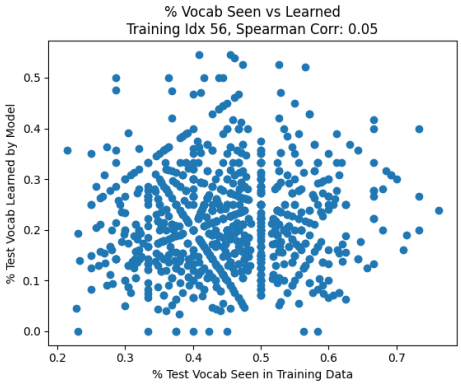

Figure 6: Plot of % Filipino vocabulary per test example trained on vs. generated at test time (i.e. learned); Training on more vocabulary does not mean the model learns to generate those vocabulary; Each point is generated using a sample in the test set

Some runs also exhibit more interactions with the model's parametric knowledge We observe that using some shuffles, models generate words not in the training data more frequently. This suggests that in some shuffles, models rely more on their parametric knowledge; hence, particular shuffles induce more reliance on parametric knowledge.

To illustrate, in one shuffle, the model correctly generates at least one OOD word in 42.7% of test samples; but only does so for 16.3% of samples using another shuffle. In Table 4, the model generates *gulang* (age/old), despite it not being in the training corpus.

Additionally, in some orderings of the data, the model incorrectly generates words from other languages more frequently, despite the training corpus solely being in Filipino. For example, it translates *he added* as *katanya*, which means "he said" in Indonesian. This happens across many test set examples[2]. In some orderings of the data, more test samples have foreign language words (Indonesian: 319, Cebuano: 232), whereas in other orderings, there are fewer (Indonesian: 154, Cebuano: 195 words). It should be noted that these numbers are overestimated as both languages share words with Filipino, but we manually review and confirm that many of them are indeed non-Filipino.

Overall, we find that some orderings lead models to learn incorrect translations, fail to acquire vocabulary, or incorrectly use words from parametric knowledge more frequently, and hence achieve worse performance, whereas others do not.

6.0.2 Some orderings negatively impact the learned representations of the data

We explore *why* certain orderings exhibit good performance, and others do not, by studying both manually crafted lexical features (See Appendix B), and the model embeddings.

We fine-tune models on various shuffles of an English-Filipino dataset, and analyze the model's predictions. We create pairs of similar examples, and analyze the behavior of the model by analyzing how the hidden state embeddings of these pairs of samples change throughout fine-tuning.

We hypothesize that some orderings of the data allow the model to learn the representation of the samples *well*, in the sense that similar sentences have similar embeddings - and hence learn sensible embeddings that yield good performance. In contrast, other orderings incorrectly "move" the

[2]We identify the languages using Python `googletrans`

Type	Text	Comment
Source	"We now have 4-month-old mice that are non-diabetic that used to be diabetic," he added	
Target	"Mayroon na tayong 4 na buwang gulang na daga na hindi diabetic na dating diabetic," dagdag niya	
Prediction (Step 23)	"We now have 4-month-old mice na hindi-diabetic," **katanya**.	Foreign (Indonesian; *katanya*: he said)
Prediction (Step 70)	"We ngayon mayroon dalawang buwan **gulang** na mga maliliit na... katawan ng"	OOD word (*gulang*: age/old)

Table 4: Models generate words not in the training data, both correctly (*gulang*) and incorrectly (*katanya*)

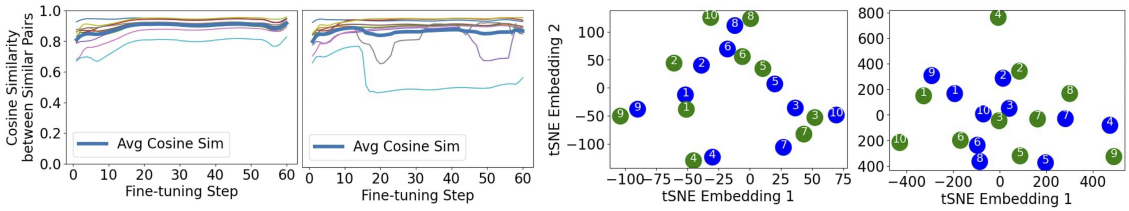

Figure 7: For the same set of data, some orderings may lead the model to learn good representations of the data, leading pairs of similar sentences to have similar embeddings (left), whereas other orderings push certain representations of sentences further away (right); For a pair of similar sentences, we compute the cosine similarity of their embeddings by step (left), and their tSNE embeddings for each pair (**green** and **blue**, same number) (right)

representations of some samples to a part of the embedding space from which they are unable to get back, thereby distorting the embedding and leading to poorer performance.

To demonstrate this, we take ten sentences from the NLLB which are in neither the train nor test set, and write a sentence with similar in meaning for each. We expect the representations of each pair to get closer as the training progresses. In some orderings (See Figure 7, left), the model learns similar representations for similar sentences, shown by the increasing cosine similarity between pairs of similar sentences, and pairs of sentences with the same IDs being close in embedding space (blue and green dots with the same ID are close). In contrast, other orderings (See Figure 7, right) incorrectly move the representation for one of the sentences further away from its pair, from which it is unable to recover, shown by the drop in cosine similarity for one pair at fine-tuning step 22, which does not go up again. Moreover, multiple pairs of sentences have dissimilar embeddings (4, 8, 9, 10).

Our findings show that some orderings of the data lead to poor acquisition of the MT task, or potentially distorted representations. This aims to motivate future work that analyzes *why* some orderings lead to suboptimal performance, and propose AL strategies that select data in way that maximizes informativeness metrics while avoiding the failure modes we observed.

7 Conclusion

In this paper, we demonstrate that applying active learning (AL) strategies to machine translation to sample data fails to consistently achieve better test set performance versus random sampling. We analyze reasons for its underperformance, and find that the ordering of samples significantly impact the model's performance. In some training runs, we observe that MT models learn distorted representations or learn wrong patterns from the data which stem from noise in the task and interactions with the model's parametric memory. By accounting for training dynamics, models can achieve better performance using data chosen using AL, and ultimately improve the use of AL in various low-resource scenarios.

This work aims to show how improving model performance is not solely a problem of optimizing for the right informativeness metrics; it requires understanding the complexities of training and learning involved in translation, and broader generation tasks. We hope these findings motivate future work in AL in text generation to explicitly consider training dynamics. Concretely, future work could (1) verify if the results generalize to other generation tasks, (2) analyze and identify interpretable characteristics of the ordering of samples that are associated with better performance to be used as heuristics in future AL algorithms, and (3) design AL strategies which select samples that are both informative and also correctly learned by the model.

Limitations

We emphasize that our results are based on very specific model and dataset choices; hence, the current results should not be taken to generalize across all tasks, datasets, and models. Moreover, we are only able to test a specific set of hyperparameters due to the computational cost of the experiments, but even the choice of hyperparameters may yield different model behaviors across runs. We also want to highlight that our section on training dynamics is based off a qualitative study of one translation direction, which the authors chose as they had access to speakers in that language. These results merely serve to provide hypotheses as to why models may fail to learn from the patterns in the data, but more rigorous experimentation is required to make stronger claims about translation or even generation as a whole.

We also note that evaluation must be done before deploying any MT model into a real world setting; while AL seeks to improve the performance of these MT models, it should by no means be naively applied and deployed without further testing.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Everlyn Asiko Chimoto and Bruce A. Bassett. 2022. COMET-QE and active learning for low-resource machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4735–4740, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

D. A. Cohn, Z. Ghahramani, and M. I. Jordan. 1996. Active learning with statistical models.

Liat Ein-Dor, Alon Halfon, Ariel Gera, Eyal Shnarch, Lena Dankin, Leshem Choshen, Marina Danilevsky, Ranit Aharonov, Yoav Katz, and Noam Slonim. 2020. Active Learning for BERT: An Empirical Study. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7949–7962, Online. Association for Computational Linguistics.

Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. *ArXiv*, abs/1703.02910.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020. The state and fate of linguistic diversity and inclusion in the NLP world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

Andreas Kirsch, Joost van Amersfoort, and Yarin Gal. 2019. Batchbald: Efficient and diverse batch acquisition for deep bayesian active learning.

Tyler LaBonte, Vidya Muthukumar, and Abhishek Kumar. 2022. Dropout disagreement: A recipe for group robustness with fewer annotations. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*.

Elite Data Labs. 2025. AI Data Annotation Costs in 2025: Pricing, Insights & Value — aidatalabelers.com. https://aidatalabelers.com/how-much-do-ai-data-annotation-services-cost-i [Accessed 17-05-2025].

Chuanming Liu and Jingqi Yu. 2023. Uncertainty-aware non-autoregressive neural machine translation. *Computer Speech Language*, 78:101444.

Tasnim Mohiuddin, Philipp Koehn, Vishrav Chaudhary, James Cross, Shruti Bhosale, and Shafiq Joty. 2022. Data selection curriculum for neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 1569–1582, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2024. Scaling neural machine translation to 200 languages. *Nature*, 630(8018):841–846.

Yotam Perlitz, Ariel Gera, Michal Shmueli-Scheuer, Dafna Sheinwald, Noam Slonim, and Liat Ein-Dor. 2023. Active learning for natural language generation. In *Proceedings of the 2023 Conference on*

Empirical Methods in Natural Language Processing, pages 9862–9877, Singapore. Association for Computational Linguistics.

Maja Popović. 2017. chrF++: words helping character n-grams. In *Proceedings of the Second Conference on Machine Translation*, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Ameya Prabhu, Charles Dognin, and Maneesh Singh. 2019. Sampling bias in deep active classification: An empirical study. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4058–4068, Hong Kong, China. Association for Computational Linguistics.

Maximilian Schmidt, A. Bartezzaghi, Jasmina Bogojeska, Adelmo Cristiano Innocenza Malossi, and Thang Vu. 2022. Combining data generation and active learning for low-resource question answering. In *International Conference on Artificial Neural Networks*.

Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*.

Aditya Siddhant and Zachary C. Lipton. 2018. Deep Bayesian active learning for natural language processing: Results of a large-scale empirical study. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2904–2909, Brussels, Belgium. Association for Computational Linguistics.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

Xiangkai Zeng, Sarthak Garg, Rajen Chatterjee, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Empirical evaluation of active learning techniques for neural MT. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 84–93, Hong Kong, China. Association for Computational Linguistics.

Ye Zhang, Matthew Lease, and Byron Wallace. 2017. Active discriminative text representation learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1).

Zhisong Zhang, Emma Strubell, and Eduard Hovy. 2022. A survey of active learning for natural language processing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6166–6190, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuekai Zhao, Haoran Zhang, Shuchang Zhou, and Zhihua Zhang. 2020. Active learning approaches to enhancing neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1796–1806, Online. Association for Computational Linguistics.

A AL Metrics

A.1 Representativeness Metrics

Delfy (Zhang et al., 2022)

$$
\begin{aligned}
&\mathrm{f}_{\text{Delfy}}(\mathcal{S}) = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}}\text{Delfy}(x)\\
&\text{Delfy}(x) =\\
&\frac{1}{|x|}\sum_{i=1}^{|x|}\frac{\log(C(x_i|U)+1)}{\sum_{w'\in U}\log(C(w'|U)+1)}\cdot p_{\text{Delfy}}(x_i)\\
&\text{lf}(x) =\\
&\frac{1}{|x|}\sum_{i=1}^{|x|}\frac{\log(C(x_i|U)+1)}{\sum_{w'\in U}\log(C(w'|U)+1)}\cdot p_{\text{Lf}}(x_i)\\
&p_{\text{Delfy}}(x_i) = e^{-\lambda_1 C(x_i|L)}\cdot e^{-\lambda_2 C(x_i|\hat{U}(x))}\\
&p_{\text{Lf}}(x_i) = e^{-\lambda_1 C(x_i|L)}
\end{aligned}
\tag{1}
$$

Where U is the set of untranslated target sentences, $\hat{U}(x)$ is the set of untranslated sentences with ls score higher than $ls(x)$, $L = \{\}$ is the (empty) set of already selected sentences, $C(w|S)$ is the number of times word w appears in a set S, and p_{Delfy} and p_{Lf} are penalty functions to penalize seen words, in which we use $\lambda_2 = 1$

L2 Distance

$$
\mathrm{f}_{\text{L2}}(\mathcal{S}) = \frac{1}{|\mathcal{S}|}\sum_{x\in\mathcal{S}}||h_{f_\theta}(x) - \bar{h}_{f_\theta}(\mathcal{S})||_2^2 \tag{2}
$$

Where $h_{f_\theta}(x) \in R^d$ is the hidden state representation of x, obtained by taking the last hidden state of encoder f_θ and averaging it over the vocab, so that it is a vector of dimension d, and $\bar{h}_{f_\theta}(\mathcal{S})$ is the average hidden state across all samples in $\mathcal{S}$.

Greedy Core Set (Sener and Savarese, 2018) We describe one round of the greedy core set by Sener and Savarese (2018) in Algorithm 2, where where $\Delta(x,y) = ||h_{f_\theta}(x) - h_{f_\theta}(y)||_2^2$

A.2 Informativeness Metrics

Average Token Probability & Entropy (Zhao et al., 2020)

$$
\mathrm{f}_{\text{ATP}}(x) = \frac{1}{|\hat{y}|}\sum_{t=1}^{|\hat{y}|}p(\hat{y}_t|\hat{y}_{<t},x) \tag{3}
$$

$$
\mathrm{f}_{\text{ATE}}(x) = \frac{1}{|\hat{y}|}\sum_{t=1}^{|\hat{y}|}\mathcal{H}\left(p(\hat{y}_t|\hat{y}_{<t},x)\right) \tag{4}
$$

Algorithm 2 Core Set Algorithm (1 Round)

Require:
$\mathcal{D}$ (Unlabeled Dataset), $\mathcal{L}$ (Labeled Dataset)
b (Budget per Round), f_θ (LM)
for $i \leftarrow 1$ to b **do**
  $u \leftarrow \text{argmax}_{x\in\mathcal{D}}\min_{y\in\mathcal{L}}\Delta(x,y)$
  $\mathcal{L} \leftarrow \mathcal{L}\bigcup\{u\}$
  $\mathcal{D} \leftarrow \mathcal{D}\backslash\{u\}$
end for

Lexical Similarity (Schmidt et al., 2022)

$$
\mathrm{f}_{\text{LS}}(x) = \frac{\sum_{i=1}^{10}\sum_{j=1}^{10}\text{Meteor}(\hat{y}^{(i)},\hat{y}^{(j)})}{N(N-1)} \tag{5}
$$

We compute lexical similarity, where similarity is measured using METEOR (Banerjee and Lavie, 2005).

BALD (Gal et al., 2017)

$$
\begin{aligned}
\mathrm{f}_{\text{BALD}}(x) &= \frac{1}{|\hat{y}|}\sum_{t=1}^{|\hat{y}|}\mathcal{H}\left(p(\hat{y}_t|\hat{y}_{<t},x)\right)\\
&-\frac{1}{k}\sum_{i=1}^{k}\frac{1}{|\hat{y}^{(i)}|}\sum_{t=1}^{|\hat{y}^{(i)}|}\mathcal{H}\left(p(\hat{y}_t^{(i)}|\hat{y}_{<t}^{(i)},x)\right)
\end{aligned}
\tag{6}
$$

$$
\begin{aligned}
&\mathcal{H}\left(p(\hat{y}_t|\hat{y}_{<t},x)\right) =\\
&\quad -\sum_{j=1}^{|\mathcal{V}|}p(\hat{y}_{t,j}|\hat{y}_{<t},x)\log\left(p(\hat{y}_{t,j}|\hat{y}_{<t},x)\right)
\end{aligned}
$$

Where $\hat{y}$ is the predicted output, $\hat{y}^{(i)}$ is the i-th predicted output generated by sampling using dropout, and $\hat{y}_t$ and $\hat{y}_t^{(i)}$ are their t-th tokens

B Analysis of Ordering Features

What about the shuffling of the data explains the variance in performance? We explore whether ordering the samples to prioritize data with specific features is associated with better performance. We first determine how well ordered the training data is with respect to a feature. To do this, we define the *slope* of a feature, which is the coefficient β obtained from regressing $feature = \alpha + \beta\cdot rank + \epsilon$ using ordinary least squares (OLS), where $rank$ is the position which a sample appears in the dataset, and $feature$ is the value of the feature for that sample. A positive β indicates that the samples are presented in increasing order of the feature, and a negative β shows decreasing order. Per sample, we define the following features:

- Lexical Features (Target): length (in words), mean word length (in characters), DelFy score
- Quality Features: translation quality score
- Model Uncertainty of the Sample: baseline model average token probability, average token entropy, BALD, and lexical similarity

To compute the translation quality of a sample, we translate the text written in the target language back into English using Google Translate[3], and compare how similar it is to the original English text using ChrF+ (Popović, 2017); a high ChrF+ indicates that the translation has good quality.

Then, to understand if ordering the samples in increasing or decreasing order with respect to a feature impacts performance, we compute the Spearman correlation between the ChrF scores and the feature slopes. A positive correlation means that training a model with samples in increasing order with respect to a feature is associated with better performance, whereas a negative coefficient means training samples in decreasing order of that feature is associated with better performance.

Ordering with respect to features of the data (e.g. length, difficulty, noise) are unable to explain the differences in performance We then study what about the ordering of the samples could explain the differences in performance. We compute the *feature slopes*, which represent how well-ordered a particular training run is with respect to a certain feature of the samples. We then compute the Spearman correlation between the ChrF scores and the feature slopes, to understand whether ordering samples by particular features is associated with better performance (See Table 5).

For Eng-Afr, training runs achieve better performance when the data is ordered from (1) short to long samples (Target Length), (2) samples which the model is certain about to those which it is uncertain about (Avg Token Entropy and Lexical Similarity), and (3) noisy to clean samples (Translation Quality). For Eng-Deu, the ordering with respect to the computed features do not have any statistically significant relationship with performance.

However, the characteristics of ordering we compute only account for 1.7% (Eng-Afr), 0.9% (Eng-Deu), 1.9% (Eng-Fil), 3.5% (Eng-Hat) of the variance in performance (computed by regressing ChrF on all feature slopes with OLS, using R^2). This suggests that while ordering *has* an impact on performance, it cannot be fully explained by easily interpretable features of the ordering.

[3]Implemented using the Python`googletrans` package

	Afr	Deu	Fil	Hat
Sample Target Length	0.029	0.022	0.071	0.047
Sample Delfy	-0.071	-0.023	0.041	0.011
Sample Translation Quality	0.029	-0.083	0.001	0.009
Model Unc. (Avg Token Prob)	0.019	0.029	-0.037	-0.003
Model Unc. (Avg Token Ent)	-0.003	-0.046	0.002	0.010
Model Unc. (BALD)	-0.003	-0.035	-0.027	0.040
Model Unc. (Lex Sim)	-0.020	0.054	0.011	-0.040

Table 5: Spearman correlation between ChrF and feature slopes, a positive/negative value means fine-tuning models with increasing/decreasing order of a feature is associated with better performance; Unc.: Uncertainty

C Fine-Tuning Details

We run all our experiments on RTX 8000 GPUs; each active learning run in the validation experiment took roughly 10 GPU hours, whereas the sampling and ordering GPU hours took roughly 72 GPU hours per translation direction.

D Dataset Details

We use the NLLB dataset (NLLB Team et al., 2024) under the ODC-By License, and the FLORES Plus dataset (Team et al., 2022) under the CC BY-SA 4.0 License, which allow the use of these datasets for research purposes. We scan the datasets to check that there are no malicious or harmful content in the translation pairs. For these datasets, we use the English-Afrikaans, English-German, English-Filipino, and English-Hatian Creole datasets.