Representative Subset Selection for Efficient Fine-Tuning in Self-Supervised Speech Recognition

Abdul Hameed Azeemi
Lahore University of Management Sciences
abdul.azeemi@lums.edu.pk

Ihsan Ayyub Qazi
Lahore University of Management Sciences
ihsan.qazi@lums.edu.pk

Agha Ali Raza
Lahore University of Management Sciences
agha.ali.raza@lums.edu.pk

Abstract

Self-supervised speech recognition models require considerable labeled training data for learning high-fidelity representations for Automatic Speech Recognition (ASR), which is computationally demanding and time-consuming. We consider the task of identifying an optimal subset of data for efficient fine-tuning in self-supervised speech models for ASR. We discover that the dataset pruning strategies used in vision tasks for sampling the most informative examples do not perform better than random subset selection on fine-tuning self-supervised ASR. We then present the COWERAGE algorithm for representative subset selection in self-supervised ASR. COWERAGE is based on our finding that ensuring the coverage of examples based on training Word Error Rate (WER) in the early training epochs leads to better generalization performance. Extensive experiments with the wav2vec 2.0 and HuBERT model on TIMIT, Librispeech, and LJSpeech datasets show the effectiveness of COWERAGE and its transferability across models, with up to 17% relative WER improvement over existing dataset pruning methods and random sampling. We also demonstrate that the coverage of training instances in terms of WER values ensures the inclusion of phonemically diverse examples, leading to better test accuracy in self-supervised speech recognition models.

1 Introduction

There has been rapid progress in recent years toward improving speech self-supervised learning (speech SSL) models. Such models learn high-fidelity speech representations using a large amount of unlabeled data and use paired data for fine-tuning on the downstream task of automatic speech recognition (ASR) [2, 6]. However, still a significant amount of labeled training data is used in the fine-tuning step for achieving robust performance, which is computationally demanding and time-consuming. For example, the standard `wav2vec2` fine-tuning procedure on Librispeech/Libri-light requires $\sim 50-100$ hours on a V100 GPU, which is significantly higher ($> 50\times$) than the cost of fine-tuning BERT on GLUE [9]. Moreover, this also hinders their usage in low-resource systems, especially compute-restricted environments (e.g., cheaper GPUs and on-device computing), which is presently a significant barrier in democratizing access to these models [1, 14].

Recent work uses adapters to enable efficient fine-tuning by using a fraction of parameters in speech SSL models [18]. However, their usage necessitates task-specific modifications, which prevents their applicability across different models and datasets. In contrast, we consider increasing the efficiency of speech SSL fine-tuning procedure by reducing training data requirements and find *smaller*, *representative* and *model-agnostic* subsets of data for fine-tuning speech SSL models.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

The data pruning mechanisms specifically tailored for deep learning models have been studied extensively for standard vision tasks. These methods focus on selecting the most informative training examples [19, 4, 14, 15, 8, 11, 12] which has been shown to perform better than the random selection of the training data. The methods for identifying the important examples in these cases are based on scores that are directly derived from the training properties and example difficulty such as the error vector norm [14], the number of times an example is forgotten during training [19] or the holdout loss [12]. However, no such mechanism has been studied yet for data pruning in speech SSL models.

We find that in standard datasets for training speech SSL models, sampling only the *hard-to-learn* training examples based on word error rate (WER) does not consistently perform better than random pruning. This is in contrast to data pruning strategies in vision tasks where this method outperforms other baselines [19, 14, 17]. For better data subset selection in fine-tuning speech SSL models, we propose COWERAGE, an algorithm designed to identify training examples important for better generalization. We find that ensuring the coverage of diverse examples based on *training WER values* in the early training epochs leads to better accuracy on unseen test data than random pruning or selecting only the most informative (hard-to-learn) examples. Experiments show the effectiveness of the COWERAGE algorithm over three primary pruning strategies: random selection, top *k* (hardest subset selection), and bottom *k* (easiest subset selection). To understand the underlying mechanism governing COWERAGE's generalization properties, we establish a connection between the training WER of the examples and their phonemic cover and find that our algorithm ensures the inclusion of phonemically diverse examples (i.e., examples of both low and high phonemic coverage) *without* explicitly learning any phoneme-level error model.

2 Method

Consider a self-supervised model $f(x;\theta)$ ($\theta \in \mathcal{R}^d$) that is pre-trained on a large unlabelled dataset $x \in \mathcal{D}_u$ on some objective $\mathcal{L}_p$. The model obtained after self-supervised pretraining with weights θ_L is then fine-tuned for the downstream task of ASR with another objective $\mathcal{L}_f$ on a labelled dataset $x \in \mathcal{D}_l$ (which is generally smaller than $\mathcal{D}_u$). $\mathcal{D}_l$ consists of transcribed audios (i.e. audio and the corresponding sentence that was uttered). Our goal is to prune $\mathcal{D}_l$ to obtain a subset B_l such that the performance of self-supervised ASR model $f(x;\theta)$ after fine-tuning on B_l is better than random pruning. We only consider pruning $\mathcal{D}_l$ (and not D_u) since we aim to directly evaluate the impact of different subset selection methods on the downstream task of ASR instead of the unsupervised pre-training of speech SSL model. The performance of an ASR model is commonly evaluated via WER ($\frac{I+D+S}{N}$), which is computed by aligning the word sequence generated by the ASR system with the actual transcription (containing *N* words) and calculating the sum of substitutions (*S*), insertions (*I*), and deletions (*D*) [21].

A number of active learning approaches are based on the inclusion of *informative* training examples in the dataset for deep learning models, i.e., examples with high error during the training epochs. We first quantify the importance of a training example in the context of a self-supervised ASR system to form a baseline for the comparison of different pruning algorithms. The training WER of an example after a few training epochs is representative of the difficulty of that example in being transcribed correctly by an ASR system. Intuitively, a hard-to-learn example will have a higher training WER due to the greater misalignment between the generated word sequence and the actual transcription. We now use the training WER to present three different subset selection strategies for selecting a subset B_l of the training data D_l for fine-tuning a self-supervised speech model on ASR.

Strategy 1: Picking the hardest *k* examples The first approach is to pick the top k training examples, i.e., the ones with the highest WER. This replicates the pruning strategy of picking the highest error examples [14, 11] during training. We first compute the training WER in a particular epoch (WER selection epoch) for all the examples. Then we select examples with the highest WER and perform fine-tuning on this subset. The number of examples selected is determined by the pruning fraction *p*.

Strategy 2: Picking the easiest *k* examples The second strategy is to pick the bottom k training examples i.e., the ones with the lowest WER. This is the inverse of strategy 1 and removes the harder-to-learn outliers from the training set in an attempt to retain representative examples.

Strategy 3: COWERAGE Subset Selection We now present a novel approach for dataset pruning, which we call COWERAGE, i.e., picking examples to ensure the *coverage* of the training WER. The following claim forms the basis of the COWERAGE algorithm.

Claim 2.1. Ensuring the coverage of training WER values guarantees the inclusion of phonemically diverse examples in the training data.

With COWERAGE, we first compute the training WER for each example in D_l, with the lowest WER as w_l and the highest WER as w_h. We then use a stratified sampling approach of partitioning N total examples from the range $[w_l, w_h]$ into M buckets, with each bucket defined as,

$$S_i = \mathcal{W}\left(w_l + \tfrac{i-1}{M}\left(w_h - w_l\right), w_l + \tfrac{i}{M}\left(w_h - w_l\right)\right)$$

where $i = 1 \ldots n$. We then use simple random sampling to select k examples uniformly from each bucket, $X_{1,\ldots,}X_k \sim \mathcal{U}\left(S_i\right)$, where k is decided by the fraction of the dataset to be pruned and the size of the bucket. $\mathcal{U}\left(S_i\right)$ denotes the uniform distribution over the set S_i. This stratified sampling method ensures coverage of WER when selecting training examples. The selected subset is used to fine-tune speech SSL model for ASR and the test performance is evaluated through WER (Fig. 1). The overall algorithm is presented in Algorithm 1.

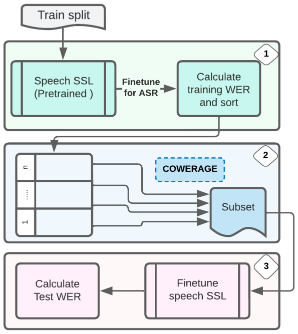

Figure 1: Fine-tuning a self-supervised model for ASR using a data subset selected by the COWERAGE algorithm.

Algorithm 1 COWERAGE Subset Selection for fine-tuning ASR Model

```
Input: SSL Pretrained Model $f$, Dataset $D_l$, Pruning Fraction $p$, Train Epoch $e$, Bucket Size $b$
$W \leftarrow$ Finetune $f$ on $D_l$ and compute WER for each example on epoch $e$
$retainFraction \leftarrow 1 - p$, $B_l \leftarrow \emptyset$
$W \leftarrow sortDescending(W)$
$buckets \leftarrow createBuckets(W, size = b)$
for $bucket$ in $buckets$ do
    $sampleSize \leftarrow retainFraction * b$
    $S \leftarrow randomSample(bucket, sampleSize)$
    $B_l \leftarrow B_l \cup S$
end for
```

2.0.1 Comparison to Random Sampling

We now highlight some key differences between random subset selection and COWERAGE.

Claim 2.2. In contrast to the COWERAGE algorithm, random sampling does not ensure selection of examples from the tail WER range. The proof is presented in Appendix A.1.

Claim 2.3. Subsets selected by COWERAGE have a lower variance of the sample mean of WER than randomly selected samples. The proof is presented in Appendix A.2.

3 Empirical Evaluation

Models and Datasets. We use the `wav2vec2-base` [2] (95M parameters) and `HuBERT-base` model [6] (90M parameters) for our experiments. We fine-tune them on the training subsets of three speech datasets: TIMIT [5], Librispeech 10h [13] and LJSpeech [7] and report WER for pruning fractions of 0.1, 0.3, 0.5, 0.7, and 0.9 to adequately evaluate low, moderate, and extreme pruning settings across different strategies. Please see Appendix D.3 for details about train and test splits and Appendix D.4 for hyperparameters.

Experiments. We fine-tune `wav2vec2-base` model on the selected dataset and calculate the WER of the training examples over ten independent runs. The training scores (averaged over 10 runs) from a particular epoch are then used to prune the examples through the pruning strategies to generate a subset of training data. The data subsets are then used to fine-tune `wav2vec2-base` and `HuBERT-base` for ASR.

Results. We show the results of pruning experiments via different strategies across multiple pruning fractions in Table 1. For each strategy and pruning fraction, we report the mean WER of three

Table 1: Test WER for the four strategies of pruning the training set evaluated at multiple pruning fractions (0.1, 0.3, 0.5, 0.7, 0.9) and different datasets (LJSpeech, LS-10h and TIMIT). The training WER in a particular epoch is averaged over 10 runs and then used for a particular pruning strategy. For each result, we do three independent runs and report the mean test WER. COWERAGE consistently demonstrates the lowest WER at various pruning fractions. WER selection epoch (WSE) is set to 8 for these experiments. See Section B.4 for WSE ablation.

Dataset	**Strategy**	`wav2vec2-base`						`HuBERT-base`					
		No pruning	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**	**No pruning**	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**
LJSpeech	Random	0.052	0.062	0.071	0.085	0.128	0.251	0.091	0.117	0.128	0.140	0.196	0.272
	Top K	0.052	0.060	0.064	0.077	0.101	0.238	0.091	0.109	0.118	0.135	0.168	0.248
	Bottom K	0.052	0.057	0.063	0.070	0.091	0.166	0.091	0.105	0.116	0.130	0.151	0.181
	COWERAGE	**0.052**	**0.054**	**0.060**	**0.067**	**0.085**	**0.144**	**0.091**	**0.101**	**0.107**	**0.115**	**0.136**	**0.153**
LS-10h	Random	0.140	0.147	0.168	0.188	0.245	0.360	0.180	0.219	0.220	0.298	0.309	0.424
	Top K	0.140	0.143	0.155	0.174	0.198	0.343	0.180	0.210	0.215	0.268	0.313	0.391
	Bottom K	0.140	0.146	0.159	0.175	0.201	0.336	0.180	0.215	0.219	0.269	0.336	0.381
	COWERAGE	**0.140**	**0.142**	**0.150**	**0.164**	**0.192**	**0.277**	**0.180**	**0.185**	**0.211**	**0.250**	**0.290**	**0.341**
TIMIT	Random	0.315	0.325	0.341	0.357	0.394	0.557	0.328	0.357	0.373	0.392	0.452	0.675
	Top K	0.315	0.322	0.334	0.392	0.472	0.678	0.328	0.345	0.366	0.435	0.532	0.871
	Bottom K	0.315	0.336	0.360	0.411	0.521	0.887	0.328	0.346	0.391	0.447	0.568	0.931
	COWERAGE	**0.315**	**0.320**	**0.333**	**0.339**	**0.369**	**0.455**	**0.328**	**0.335**	**0.355**	**0.381**	**0.445**	**0.616**

independent runs. We observe that for the majority of pruning fractions, COWERAGE subset selection is consistently better than the other three pruning strategies (top *k*, bottom *k*, and random pruning) for TIMIT, LS-10h, and LJSpeech. At higher pruning fractions, the difference between the test WER for COWERAGE and the other pruning strategies increases, e.g., on the Librispeech-10h dataset with 90% pruning, COWERAGE shows 17% relative WER improvement over Bottom K strategy compared to 5% relative WER improvement at 30% pruning. This observation can also be made for random sampling and is consistent with claim 2.2 where we consider the impact of smaller sample sizes (higher pruning percentages) on the selection of examples from tail WER which subsequently affects test error. On the TIMIT dataset, going from 10% pruning to 90% pruning leads to an absolute increase of only 0.135 WER for COWERAGE compared to an increase of 0.551, 0.356, and 0.232 for Bottom K, Top K, and Random respectively.

Transferability of representative subsets. Table 1 shows that COWERAGE demonstrates better performance in the fine-tuning run of `HuBERT-base` on the subsets constructed through training WER values of `wav2vec2-base`. The relative trend for other pruning strategies is also similar to that of `wav2vec2-base`. This suggests that the representative subsets computed through one speech SSL model are *transferable* to another speech SSL model, making them *model-agnostic* and *dataset-specific*. This property is present in a few other pruning metrics for deep learning models as well, including EL2N score [14] and RHO-loss [12]. Our explanation is that since the composition of the representative subset is more influenced by the *ranking* of training examples instead of absolute WER values, it makes them relevant for fine-tuning other speech SSL models. Additionally, the prior averaging of the training WER values theoretically eliminates the influence of specific model weights, which produces a more precise ranking of the examples. We can consider the representative subsets constructed through COWERAGE as *foundation datasets* [17] which need to be constructed once and can be later used to fine-tune multiple other speech SSL models.

Connection to Phonemes. To verify claim 2.1 and understand why COWERAGE performs better than other pruning strategies, we conduct an experiment to determine how the phoneme distribution of training examples varies with the training WER. We find an inverse relationship between the training WER and the phonemic cover and identify that the coverage of training WER values in a particular subset leads to the inclusion of phonemically diverse training examples (see Appendix C).

4 Conclusion

In this work, we proposed COWERAGE, a new method for pruning data for self-supervised automatic speech recognition, which relies on sampling data in a way that ensures coverage of training WER values. An evaluation on `wav2vec2` and `HuBERT` and three datasets show that COWERAGE performs better than random selection and other data pruning strategies that select harder-to-learn or easier-to-learn examples. While we designed our approach to be dataset agnostic and applicable to different distributions of training WER, it remains to be empirically evaluated whether our methodology generalizes to noisier data and multilingual speech corpora.

References

[1] Nur Ahmed and Muntasir Wahed. "The de-democratization of ai: Deep learning and the compute divide in artificial intelligence research". In: *arXiv preprint arXiv:2010.15581* (2020).

[2] Alexei Baevski et al. "wav2vec 2.0: A framework for self-supervised learning of speech representations". In: *arXiv preprint arXiv:2006.11477* (2020).

[3] Rosey Billington, Hywel Stoakes, and Nick Thieberger. "The Pacific Expansion: Optimizing Phonetic Transcription of Archival Corpora". In: *Proc. Interspeech 2021*. 2021, pp. 4029–4033. DOI: `10.21437/Interspeech.2021-2167`.

[4] Cody Coleman et al. "Selection via proxy: Efficient data selection for deep learning". In: *arXiv preprint arXiv:1906.11829* (2019).

[5] John S Garofolo et al. "DARPA TIMIT acoustic-phonetic continous speech corpus CD-ROM. NIST speech disc 1-1.1". In: *NASA STI/Recon technical report n* 93 (1993), p. 27403.

[6] Wei-Ning Hsu et al. "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units". In: *arXiv preprint arXiv:2106.07447* (2021).

[7] Keith Ito and Linda Johnson. *The lj speech dataset*. 2017.

[8] Siddharth Karamcheti et al. "Mind your outliers! investigating the negative impact of outliers on active learning for visual question answering". In: *arXiv preprint arXiv:2107.02331* (2021).

[9] Cheng-I Jeff Lai et al. "PARP: Prune, Adjust and Re-Prune for Self-Supervised Speech Recognition". In: *arXiv preprint arXiv:2106.05933* (2021).

[10] Alexander H Liu et al. "Towards End-to-end Unsupervised Speech Recognition". In: *arXiv preprint arXiv:2204.02492* (2022).

[11] Katerina Margatina et al. "Active Learning by Acquiring Contrastive Examples". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 650–663.

[12] Sören Mindermann et al. "Prioritized training on points that are learnable, worth learning, and not yet learnt". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 15630–15649.

[13] Vassil Panayotov et al. "Librispeech: an asr corpus based on public domain audio books". In: *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2015, pp. 5206–5210.

[14] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. "Deep Learning on a Data Diet: Finding Important Examples Early in Training". In: *Advances in Neural Information Processing Systems* 34 (2021).

[15] Ravi S Raju, Kyle Daruwalla, and Mikko Lipasti. "Accelerating Deep Learning with Dynamic Data Pruning". In: *arXiv preprint arXiv:2111.12621* (2021).

[16] Kyuhong Shim, Jungwook Choi, and Wonyong Sung. "Understanding the role of self attention for efficient speech recognition". In: *International Conference on Learning Representations*. 2021.

[17] Ben Sorscher et al. "Beyond neural scaling laws: beating power law scaling via data pruning". In: *arXiv preprint arXiv:2206.14486* (2022).

[18] Bethan Thomas, Samuel Kessler, and Salah Karout. "Efficient Adapter Transfer of Self-Supervised Speech Models for Automatic Speech Recognition". In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 7102–7106.

[19] Mariya Toneva et al. "An empirical study of example forgetting during deep neural network learning". In: *arXiv preprint arXiv:1812.05159* (2018).

[20] Thomas Wolf et al. "Huggingface's transformers: State-of-the-art natural language processing". In: *arXiv preprint arXiv:1910.03771* (2019).

[21] JP Woodard and JT Nelson. "An information theoretic measure of speech recognition performance". In: *Workshop on standardisation for speech I/O technology, Naval Air Development Center, Warminster, PA*. 1982.

Supplementary Material

A Proofs

A.1 Proof of Claim 3.2

Proof. We consider the probability of randomly selecting an example WER (w) that is at least at a distance of k standard deviation σ from the mean WER. By Chebyshev's inequality: $\Pr(|X - \bar{W}| \geq k\sigma) \leq \frac{1}{k^2} = p$, which demonstrates that increasing the WER boundary w (and hence k) decreases the probability of randomly selecting a sample with WER greater than w.[1] We now consider the probability of having at least one sample with a WER greater w when we independently draw n samples from the training WER distribution. This is a complement of the event *no sample having a WER greater than w in n draws* which is $(1 - p)^n$, and hence the event of interest has the probability upper bound $1 - (1 - p)^n = 1 - (1 - \frac{1}{k^2})^n$. This demonstrates that decreasing the sample size and increasing the pruning percentage reduces the probability of selecting a tail WER example. In contrast, for COWERAGE, the probability of selecting at least one example with a WER greater than $\bar{W} + k\sigma$ is $Pr(|S_i| > 0) = q$, where S_i is a tail bucket with the WER range (a, b) such that $a \geq \bar{W} + k\sigma$ and $b > a$. This probability (q) approaches 1 if we consider a bucket size satisfying the range (a, b), and hence COWERAGE ensures selection of examples from the tail WER range. □

A.2 Proof of Claim 3.3

Proof. We first consider the variance of samples selected by COWERAGE. Let S_{ij} be the sample i from bucket S_j. The average WER in bucket j is $\bar{W}_j = \frac{\sum_i S_{ij}}{k}$, variance in bucket j is σ_j^2 and the overall average is $\bar{W} = \frac{\sum_j \bar{W}_j}{M}$. The variance of the sample mean of WER is,

$$\mathrm{Var}_{\text{COWERAGE}}[\bar{W}] = \frac{\sum_j \mathrm{Var}\left[\bar{W}_j\right]}{M^2} \tag{1}$$

$\mathrm{Var}\left[\bar{W}_j\right]$ is the variance of the sample mean within a particular bucket and is equivalent to $\frac{\sigma_j^2}{k}$. Thus, we get

$$\mathrm{Var}_{\text{COWERAGE}}[\bar{W}] = \frac{\sum_j \mathrm{Var}\left[\bar{W}_j\right]}{M^2} = \frac{\sum_j \sigma_j^2}{M^2 k} = \frac{\sum_j \sigma_j^2}{MN} \tag{2}$$

Now we consider the variance of a simple random sample. $\mathrm{Var}[\bar{W}] = \frac{\sigma^2}{N}$ with $\sigma^2 = \mathbb{E}\left[W^2\right] - \mu^2$. Considering the contribution from each bucket in the random sample, we can specify $\sigma^2 = \frac{\sum_j \mathbb{E}[S_j]}{M} - \mu^2 = \frac{\sum_j \left(\mu_j^2 + \sigma_j^2\right)}{M} - \mu^2 = \frac{\sum_j \left((\mu_j - \mu)^2 + \sigma_j^2\right)}{M}$. Thus,

$$\mathrm{Var}_{\text{RANDOM}}[\bar{W}] = \frac{\sum_j \left((\mu_j - \mu)^2 + \sigma_j^2\right)}{MN} \tag{3}$$

Comparing (1) and (3), $\mathrm{Var}_{\text{RANDOM}}[\bar{W}] \geq \mathrm{Var}_{\text{COWERAGE}}[\bar{W}]$ and the result follows. □

[1] Note that the probability of sampling from the tail of the WER degrades *quadratically*.

B Ablation study

The Impact of Offset. To identify whether there is another contiguous subset of examples below the ones with the highest WER which can perform better than random pruning, we introduce an offset while selecting the top *k* training examples, mirroring the protocol presented by [14]. We compute the training WER for the examples and sort them in ascending order. We then maintain a sliding window from offset k to $k + N$ which keeps N data points but incrementally excludes the training examples with the highest WER. For offset sizes from 0 to 500, we notice a change in the test WER but no single offset size is consistently better than random pruning. An important implication of this finding is that no contiguous subset of training examples picked according to the WER is better than random pruning in the TIMIT speech corpus, contrary to the previous studies on vision datasets that have shown a clear correlation between the top-scoring examples and the accuracy [14].

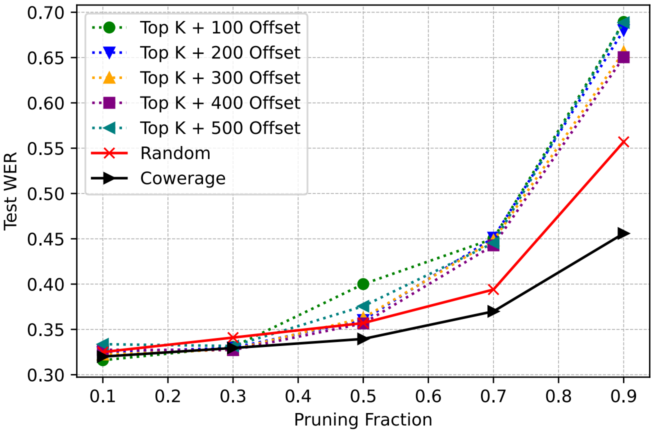

Figure 2: The test WER for the different offsets while picking the top *k* examples compared over different pruning fractions of the TIMIT dataset. Note that no single offset consistently performs better than random pruning.

B.1 Selection within the buckets.

The strategy proposed in the original COWERAGE algorithm is to randomly sample elements from each bucket. We also evaluate two other strategies: picking the first *k* examples within each bucket and picking the last *k* ones, similar to strategies 1 and 2 except that now we are sampling within a particular bucket. The results in Table 2 show that the random selection outperforms other strategies. Additionally, we evaluate the impact of increasing the bucket size on the test WER in Appendix B.5.

Table 2: Test WER for different strategies of picking samples within each bucket for COWERAGE algorithm on 0.7 pruning fraction and WER Selection Epoch 8.

	COWERAGE + Top *k*	COWERAGE + Bottom *k*	COWERAGE + Random
WER	0.378 ± 0.002	0.401 ± 0.002	0.369 ± 0.004

B.2 Phoneme Recognition on TIMIT

We evaluate the subset selection methods on the task of phoneme recognition with `wav2vec2-base` on TIMIT dataset and report the phoneme error rate (PER) on the test set (Table 3). COWERAGE consistently demonstrates the lowest PER on all the pruning fractions above 0.2.

B.3 Training time for subsets

Practically, the choice of pruning fraction can be made according to the intended size of the final dataset under the given time and memory constraints. We conduct an experiment to determine the

Table 3: Phoneme recognition on the TIMIT dataset with `wav2vec2-base`. We report PER for multiple pruning fractions and different strategies.

Strategy	Pruning Fraction				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
Random	0.124	0.133	0.148	0.230	1.000
Top K	**0.118**	0.137	0.168	0.244	1.000
Bottom K	0.122	0.142	0.170	0.282	1.000
COWERAGE	0.120	**0.133**	**0.145**	**0.211**	1.000

total steps required for convergence and the real training time for `wav2vec2` on TIMIT. The results are shown in Table 4 (for a constant learning rate). We report the real training time for the pruned datasets as a fraction of the training time for the complete dataset (x) for relative comparison. There is a significant reduction in training time for higher pruning fractions.

Table 4: Steps required for convergence and training time for `wav2vec2` on TIMIT for different pruning fractions. We replicate the results of COWERAGE from Table 1 for relative comparison.

Pruning Fraction	0.9	0.7	0.5	0.3	0.1	0
Steps required for convergence	1050	1900	2400	2800	3170	3350
Training time	0.42×	0.62×	0.77×	0.85×	0.90×	×
Test WER (COWERAGE)	0.455	0.369	0.339	0.333	0.320	0.315

B.4 WER Selection Epoch

An important hyperparameter in the COWERAGE algorithm is the epoch at which the training WER is computed for individual examples and then used for pruning i.e. the WER selection epoch. We evaluate the effect of different selection epochs on the final test WER (Table 5) in TIMIT and observe that the training WER in the early training epochs can be reliably used for ranking the examples and applying a particular pruning strategy. Hence, we select WSE = 8 for the final results in Table 1. Note that COWERAGE consistently demonstrates a lower WER than other strategies on *all epochs* that we test (8, 12, 16, 20) for the majority of pruning fractions ($0.2 - 0.9$) across all the datasets (TIMIT, LS-10h, LJSpeech). This suggests that the selection of a reasonable WSE can usually be made with less than five distinct epoch values while still achieving better results than the other strategies.

B.5 Selecting the number of buckets

We conduct an experiment with different bucket sizes on `wav2vec2` and TIMIT with 0.7 pruning fraction. The results are shown in Table 6. Our evaluation shows that increasing the bucket size beyond a certain threshold provides diminishing returns in performance. Increasing the bucket size from 50 to 100 yielded 4.8% reduction in WER whereas increasing it from 100 to 500 resulted in only a 0.27% reduction in WER.

Choosing 500 buckets in the COWERAGE algorithm provided robust performance across a wide range of dataset sizes, which ranged from 4620 examples in TIMIT to more than 10,000 examples in LJSpeech. The number of buckets can be increased further but it should be no greater than `pruningFraction * datasetSize`.

B.6 Transferability to larger models

To find out if the subsets created through a smaller model are transferable to a larger speech SSL model, we conduct an experiment with `wav2vec2-large` (317M parameters; pre-trained on Librispeech 960h) and fine-tune it on the subsets constructed through `wav2vec2-base`. We observe that COWERAGE subsets still outperform the rest of the pruning strategies, further validating the hypothesis of transferability of pruning scores.

Table 5: Test WER for the four strategies of pruning the training set evaluated at multiple pruning fractions and different training WER selection epochs. The training WER in a particular selection epoch is averaged over 10 runs and then used for a particular pruning strategy. For each result, we do three independent runs and report the mean test WER. COWERAGE consistently demonstrates the lowest WER at various pruning fractions and selection epochs. WSE: WER Selection Epoch.

WSE	Strategy	Pruning Fraction					
		No pruning	**0.1**	**0.3**	**0.5**	**0.7**	**0.9**
8	Random	0.315	0.325	0.341	0.357	0.394	0.557
	Top K	0.315	0.322	0.334	0.392	0.472	0.678
	Bottom K	0.315	0.336	0.360	0.411	0.521	0.887
	COWERAGE	0.315	**0.320**	**0.333**	**0.339**	**0.369**	**0.455**
12	Random	0.315	0.325	0.341	0.357	0.394	0.557
	Top K	0.315	0.316	0.345	0.386	0.461	0.579
	Bottom K	0.315	0.323	0.353	0.398	0.499	0.781
	COWERAGE	0.315	**0.322**	**0.328**	**0.354**	**0.370**	**0.536**
16	Random	0.315	0.325	0.341	0.357	0.394	0.557
	Top K	0.315	0.324	0.332	0.413	0.467	0.704
	Bottom K	0.315	0.323	0.346	0.382	0.468	0.657
	COWERAGE	0.315	**0.322**	**0.329**	**0.356**	**0.382**	**0.565**
20	Random	0.315	0.324	0.340	0.357	0.401	0.557
	Top K	0.315	0.328	0.370	0.422	0.518	0.709
	Bottom K	0.315	0.321	0.352	0.389	0.457	0.587
	COWERAGE	0.315	**0.321**	**0.334**	**0.340**	**0.376**	**0.545**

Table 6: Test WER for `wav2vec2` on TIMIT for different number of buckets in the COWERAGE algorithm

Number of Buckets	1	10	50	100	500	1000
Test WER	0.394	0.393	0.389	0.370	0.369	0.369

Table 7: Test WER for different for `wav2vec2-large` fine-tuned on subsets created through `wav2vec2-base`.

Strategy	Pruning Fraction				
	0.1	**0.3**	**0.5**	**0.7**	**0.9**
Random	0.300	0.308	0.322	0.356	0.545
Top K	0.295	0.297	0.345	0.385	0.634
Bottom K	0.306	0.326	0.391	0.505	0.833
COWERAGE	**0.290**	**0.296**	**0.318**	**0.332**	**0.490**

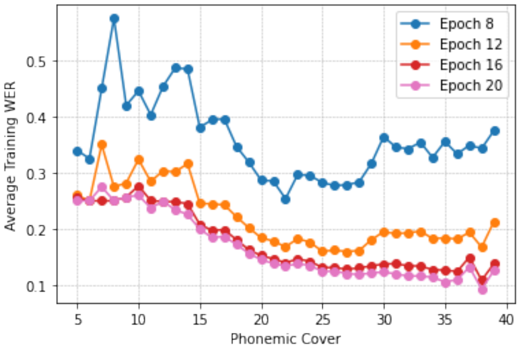

Figure 3: The training WER and the phonemic cover of examples in TIMIT dataset (without pruning) compared over multiple training epochs. The WER is computed by averaging the training scores of the examples with the same phonemic cover. The training scores for each training example and a particular epoch are computed by averaging over 10 runs.

B.7 Standard deviation for test WER on TIMIT

Table 8: The standard deviation for the test WER of `wav2vec2` presented in Table. 5

WSE	Strategy	Pruning Fraction				
		0.1	**0.3**	**0.5**	**0.7**	**0.9**
TIMIT	Random	±0.003	±0.005	±0.002	±0.025	±0.003
	Top K	±0.001	±0.007	±0.010	±0.001	±0.002
	Bottom K	±0.002	±0.002	±0.002	±0.009	±0.002
	COWERAGE	±0.001	±0.006	±0.016	±0.004	±0.005

C Connection to Phonemes

To understand why COWERAGE performs better than other pruning strategies, it is important to find out how does the phoneme distribution of training examples vary with the training error during fine-tuning of the self-supervised speech recognition models. We now perform empirical analysis to verify claim 2.1. For this analysis, we select the standard TIMIT dataset as it contains time-aligned, hand-verified phonetic and word transcriptions for each training example.

We first record the training WER of each training example in the TIMIT dataset over 10 runs and average it. Then, we compute the total number of unique phonemes in each example, which we call the *phonemic cover*. Subsequently, we group together the training examples with same phonemic cover and calculate the average training WER for each group (Fig. 3). In the earlier training epochs, the examples with a relatively low (< 17) or a high (> 28) phonemic cover have a greater WER (blue line in Fig. 3) as compared to the examples with a moderate number of phonemes ($17 \leq \texttt{phonemicCover} \leq 28$). In the later epochs ($\geq 12$), the inverse relationship between the training WER and the phonemic cover becomes more evident; the examples with a greater number of distinct phonemes have a lower training WER and vice versa.

Significance. This relationship between the training WER and the phonemic cover has several implications. Firstly, it demonstrates that there is a sizable population of sentences with a low phonemic cover that are harder to learn and hence represent a high training WER. Similarly, there are many low WER sentences with a high phonemic cover. More importantly, this experiment validates our claim that ensuring the coverage of training WER values in a particular subset leads to the inclusion of phonemically diverse training examples *without* explicitly learning any phoneme-level error model. This is beneficial as accurate phonetic data is not available for the majority of 7000 spoken languages [3]. In contrast, any method that directly ensures phoneme diversity requires an accurate phonetic transcription beforehand, which is a resource-intensive process requiring manual labeling by linguists.

To verify if the difference between the phoneme distributions of the examples within the COWERAGE subset and the other two strategies (top *k* and bottom *k*) is statistically significant, we conduct the Mann-Whitney U test, a non-parametric test, at a significance level of 0.01. We found that the differences were statistically significant at the 1% level (p-value < 0.01). The results are shown in Table 9.

Table 9: The statistical significance of the difference between the phoneme distribution of the examples within the COWERAGE subset and the other two strategies (top *k* and bottom *k*). MWU: Mann-Whitney U.

	MWU	p-value
Top *k* vs COWERAGE	2146027.5	< 0.001
Bottom *k* vs COWERAGE	2229653.0	< 0.001

C.1 Phonemic diversity and latent representation in speech SSL

How does phonemic diversity impact the discrete latent speech representations within self-supervised speech recognition models? To answer this, we study the latent representation ($\mathbf{q}_t$) learned by the quantizer within `wav2vec2` for different phonemes. [2] analyze the conditional probability $P\left(phoneme \mid \mathbf{q}_t\right)$ for each of the 39 phonemes in the TIMIT train set by computing the co-occurence between the phonemes and speech latents (see Appendix D of [2]). They demonstrate that different discrete latents specialize in different phonetic sounds in `wav2vec2` model. Building upon this, [16] analyze the relationship between attention and phonemes in Transformer-based ASR models by considering the attention map that extracts phonologically meaningful features. They observe that the characteristic feature of phonetic localization is the higher attention weights assigned to similar phonemes in the attention map (see Fig. 3 of [16]). Given these observations, we hypothesize that the performance gains for COWERAGE are due to the greater phonemic diversity which enables a more robust latent representation of each phoneme in `wav2vec2`. This view is supported by the results in Table 1 which demonstrate bigger gains in test WER for higher pruning fractions in COWERAGE. We conjecture that this is due to greater example diversity provided by COWERAGE and lack of representation of examples from the tail WER range in the case of other approaches.

D Implementation Details

D.1 Resources

We use a single 80GB NVIDIA A100 GPU for running all the experiments on the cloud. In this setting, the standard `wav2vec2-base` fine-tuning step (single run) on multiple pruning fractions took ≈ 1.25 GPU hours for the TIMIT dataset, ≈ 6 GPU hours for LJSpeech dataset, and ≈ 5.5 GPU hours for Librispeech 10h dataset. The total project (from the early experiments to the final results) consumed about 2200 GPU hours.

D.2 Code and Licenses

We release our code under the MIT license. All the data pruning strategies are implemented in Python, and the resulting subsets are used to fine-tune `wav2vec2`. The publicly available HuggingFace [20] implementation [1] of `wav2vec2-base` model[2] is used which is based on the standard `wav2vec2-base-960h` fairseq implementation[3]. The HuggingFace transformers repo is available under the Apache License 2.0 license and the fairseq repo is available under the MIT license.

D.3 Data

TIMIT [5]. We use the full TIMIT dataset with predefined training and test sets. The training set contains 4620 examples and the test set contains 1680 examples. TIMIT is available under the LDC

[1] https://github.com/huggingface/transformers

[2] https://huggingface.co/facebook/wav2vec2-base-960h

[3] https://github.com/pytorch/fairseq/blob/main/examples/wav2vec/README.md

User Agreement for Non-Members.

Librispeech [13]. We construct Librispeech 10h fine-tuning split by selecting 10h of utterances randomly from the 100h train-clean split. The test-clean split is used for evaluation. Librispeech is available under the CC BY 4.0 license.

LJSpeech [7]. This dataset contains 24 hours of English speech from a single speaker. For validation and testing, we randomly select 300 utterances, mirroring the protocol followed in earlier works [10]. The rest is used for training. LJSpeech is available under the public domain license.

D.4 Training

In all experiments, `wav2vec2-base` is fine-tuned with a batch size = 8, epochs = 20, mean ctc-loss-reduction, weight decay 0.005, and FP16 training. We use a data collator to pad the inputs dynamically. For calculating the WER for each training example, we run a computation step after each epoch and record the WER. The training WER in each epoch is averaged over 10 runs and then used for a particular pruning strategy. For each test WER reported, we do three separate runs with independent model initialization. A bucket size of 500 is chosen for the COWERAGE strategy, which is sufficiently small to ensure the selection of representative examples for different pruning fractions.