# Efficient Supernet Training with Orthogonal Softmax for Scalable ASR Model Compression

Jingjing Xu[*†‡], Eugen Beck[*‡], Zijian Yang[†‡], Ralf Schlüter[†‡]

[†]Machine Learning and Human Language Technology Group, RWTH Aachen University, Germany

[‡]AppTek GmbH, Germany

*Abstract*—ASR systems are deployed across diverse environments, each with specific hardware constraints. We use supernet training to jointly train multiple encoders of varying sizes, enabling dynamic model size adjustment to fit hardware constraints without redundant training. Moreover, we introduce a novel method called *OrthoSoftmax*, which applies multiple orthogonal softmax functions to efficiently identify optimal subnets within the supernet, avoiding resource-intensive search. This approach also enables more flexible and precise subnet selection by allowing selection based on various criteria and levels of granularity. Our results with CTC on *Librispeech* and *TED-LIUM-v2* show that FLOPs-aware component-wise selection achieves the best overall performance. With the same number of training updates from one single job, WERs for all model sizes are comparable to or slightly better than those of individually trained models. Furthermore, we analyze patterns in the selected components and reveal interesting insights.

*Index Terms*—Speech recognition, Supernet training, CTC, FLOPs-aware.

## I. INTRODUCTION

Balancing accuracy and latency in automatic speech recognition (ASR) is a critical and active area of research. The quest for higher accuracy has driven the development of larger, more complex models [1]–[3], resulting in increased latency and higher energy consumption. This trade-off is especially challenging in resource-constrained environments like client-edge devices, where computational power (measured in floating-point operations per second, or FLOPS) and memory (the number of parameters) are limited [4].

To reduce the model size with minimal loss of performance, model compression techniques such as pruning [5]–[8], knowledge distillation [9]–[11], quantization [12], [13] are commonly used in ASR. Quantization is orthogonal to other methods and can be applied to almost any model. However, the first two methods typically require a converged base model, followed by separate fine-tuning for each compressed model of varying sizes, causing redundant training and re-optimization efforts [14]. Hence, another line of methods—supernet training, first proposed in [15]—aims to jointly train a supernet encompassing multiple subnets with a single training process. In this way, the supernet can generalize across various subnet configurations, being adaptable to different model sizes.

Recent works [14], [16]–[20] in ASR have explored the concept of supernet training and achieved promising results. The challenge, however, remains in determining the optimal

* Denotes equal contribution

subnets efficiently from the supernet. The authors of [16]–[18] identify subnets by manually selecting the bottom layers or layers at regular intervals. The authors of [14], [19] use evolutionary search to find the most performant subnets from a predefined search space at each training step. Meanwhile, the authors of [20], [21] conduct search after training. In both cases, the search process is computationally expensive. Recently, this issue is addressed in [22] by combining pruning with supernet training to automatically learn the optimal subnets during training. However, the limitations are: 1) selection is restricted to entire layers, 2) the subnet size is determined by the number of layers, without considering other criteria such as number of parameters, 3) the use of straight through estimator (STE) [23] introduces gradient inconsistency, causing training instability and degraded performance [24].

Therefore, in this work, we propose a method called *OrthoSoftmax*, which apply multiple softmax functions to independent, learnable score vectors. We also introduce orthogonal constraint to let each softmax only select one distinct group of parameters. The softmax results are then aggregated to generate binary masks that determine the subnets. During training, this approach begins with a uniformly distributed soft mask and then progressively converges to a binary hard mask for each subnet. Compared to other commonly used binary mask learning approaches, this method avoids the approximation inaccuracies commonly associated with continuous relaxations of discrete operations, further details are discussed in Sec III-B. We also extend [22] by enabling finer selection granularity and introducing additional selection criteria, which facilitates more precise and flexible subnet selection. We evaluate our approach by conducting experiments with the Conformer [25] connectionist temporal classification (CTC) [26] model on both *Librispeech* and *TED-LIUM-v2* datasets. The results show that FLOPs-aware component-wise selection performs best, yielding models that achieve WERs on par with individually trained ones while significantly reducing total training time. We also analyze the patterns in component selection for subnets of varying sizes in Sec III-F.

## II. EFFICIENT SUPERNET TRAINING PIPELINE

This section explains how we use supernet training to generate one supernet and $M$ subnets with fully shared parameters across varying sizes. Let the acoustic encoder have learnable parameters $\boldsymbol{\theta}$, decomposed into $N$ parameter groups $\{\theta_j\}_{j=1}^{N}$. The costs for each group are $\boldsymbol{c} =$

$\{c_j\}_{j=1}^N$. Each of the $M$ subnets has a different total cost constraint, denoted as $\{\tau_1, \tau_2, \ldots, \tau_M\}$. For each subnetwork $n_m$, we learn a binary mask $\boldsymbol{z_m} \in \{0,1\}^N$, where parameter group $\theta_j$ is retained in the encoder $n_m$ if $z_m^j = 1$. Therefore, we can formulate the joint training problem as:

$$\min_{\theta, \{z_m\}_{m=1}^M} \mathbb{E}_{(x,y)\in\mathcal{D}} \left[ \mathcal{L}_{ASR}(x,y;\theta) + \sum_{m=1}^M \lambda_m \mathcal{L}_{ASR}(x,y;\theta \odot z_m) \right] \quad (1)$$

$$\text{subject to } \sum_{j=1}^N z_m^j \cdot c_j < \tau_m, \forall m \in \{1,\ldots,M\}, \quad (2)$$

where $\mathcal{D}$ is the training data, $\odot$ denotes the element-wise product. To better balance the loss scales between different encoders, we apply focal loss [27] as in [28] to compute $\lambda_m$ (we set $\beta_{focal}$ to 1), where p denotes the model posterior of target sequence $y$ given input sequence $x$:

$$\lambda_m = [1 - p(y|x, \theta \odot z_m)]^{\beta_{focal}} \quad (3)$$

Our efficient training pipeline consists of two steps: Step 1 aims at learning masks $\boldsymbol{z_m}$ for each subnet $n_m$. Step 2 jointly trains the supernet and subnets determined by $\boldsymbol{z_m}$ to further enhance WER. We allocate 60% of the total training steps to Step 1 and 40% to Step 2, as recommended by [22].

### A. Step 1 - Learning Subnet Binary Mask

To enable smooth adjustment and gradient-based optimization of component inclusion, we propose *OrthoSoftmax*. This method uses $N$ softmax functions to compute a soft mask $\boldsymbol{z_m} \in [0,1]^N$, where each value reflects the importance of different parameter groups. We introduce a learnable score matrix $\boldsymbol{S} \in \mathbb{R}^{N \times N}$, initialized to zeros. Applying $N$ softmax functions to the rows of $\boldsymbol{S}$ produces the weight matrix $\boldsymbol{W} \in [0,1]^{N \times N}$, where $\boldsymbol{W}_{i,j} = \text{softmax}_i(\boldsymbol{S}_{i,j}) = \frac{e^{\boldsymbol{S}_{i,j}/T}}{\sum_{j=1}^N e^{\boldsymbol{S}_{i,j}/T}}$, $\text{softmax}_i$ denotes the $i^{th}$ softmax. Temperature $T$ is annealed from 1 to a minimum of 0.1 by factor of 0.999992 at each step. Assuming the maximum number of parameter groups for subnetwork $n_m$ that fit the cost constraint $\tau_m$ is $k_m$, we design the training to achieve the following objectives for the $\boldsymbol{W}$:

1) Orthogonality: Ensure all row pairs in $\boldsymbol{W}$ are orthogonal. This will force each softmax selects exactly one non-overlapping parameter group.
2) Importance Ranking: Ensure that the parameter groups selected from the upper rows are more critical to ASR performance than those selected from the lower rows.

we compute $\boldsymbol{z_m} = \sum_{i=1}^{k_m} \boldsymbol{W}_{i,:}$, $\boldsymbol{z_m}$ initially starts with a uniform distribution and will naturally converge to a $k_m$-hot vector during the training process.

**Determine $k_m$ for Each Subnetwork $n_m$.** For each training step, we compute $k_m$ that satisfies the cost constraint $\tau_m$:

$$k_m = \max\{k\} \text{ s.t. } \sum_{i=1}^k \left( \sum_{j=1}^N \boldsymbol{W}_{ij} \cdot c_j \right) < \tau_m \quad (4)$$

During training, $k_m$ will adjust according to $\boldsymbol{W}$. As each row $\boldsymbol{W}_{i,:}$ approaches a one-hot vector, $k_m$ becomes more reliable.
**Orthogonality.** The orthogonal constraint helps achieve the design objective 1. Orthogonal constraints have been used in other work [29], [30] to improve the numerical stability of matrices. Assume $k_m$ is computed in the current training step. We apply the orthogonal constraint to the top $k_m$ rows of the matrix $\boldsymbol{W}$. For all $i, j \in \{1, \ldots, k_m\}$ with $i \neq j$, we expect $\boldsymbol{W}_{i,:} \perp \boldsymbol{W}_{j,:}$, meaning $\boldsymbol{W}_{i,:} \cdot \boldsymbol{W}_{j,:} = 0$. Suppose the product of sub-matrix $\boldsymbol{W}_{1:k_m,:}$ and its transpose $\boldsymbol{W}_{1:k_m,:}^\intercal$ is $\boldsymbol{D} \in \mathcal{R}^{k_m \times k_m}$. We aim to force $\boldsymbol{D}$ to be close to the identity matrix $\boldsymbol{I} \in \{0,1\}^{k_m \times k_m}$ by minimizing the Frobenius norm of their difference. Since $\boldsymbol{D}$ is symmetric, we only take its upper triangle part, $\boldsymbol{D}_{ut} = [\boldsymbol{D}_{ij} \mid i \leq j]$ for loss computation:

$$\mathcal{L}_{orthog} = \|\boldsymbol{D}_{ut} - \boldsymbol{I}\|_F = \sqrt{\sum_{i=1}^{k_m}(D_{i,i}-1)^2 + \sum_{i=1}^{k_m}\sum_{j=i+1}^{k_m} D_{i,j}^2} \quad (5)$$

**Importance Ranking.** Since we select rows from top to bottom, matrix $\boldsymbol{W}$ is forced to place parameter groups that are more critical for ASR performance in upper rows. To elaborate, $\tau_m < \tau_{m'}$ implies $k_m < k_{m'}$, the parameter groups selected for subnet $n_{m'}$ encompass those selected for subnet $n_m$. The $k_m$ parameter groups for subnet $n_m$ should be the most important among $k_{m'}$ parameter groups for subnet $n_{m'}$.
*1) Training Loss:* The overall training loss of Step 1 can be formulated as:

$$\mathcal{L}_{ASR}(x,y;\theta) + \lambda\mathcal{L}_{ASR}(x,y;\theta \odot z_m) + \beta\mathcal{L}_{orthog}(\boldsymbol{W}_{1:k_m,:}), \quad (6)$$

where $m$ is randomly selected from the set $\{1, \ldots, M\}$ at each training step. Compared to Eq.1, Eq.6 trains only the supernet with one subnet to avoid scaling the training cost with the number of subnets but still maintain overall performance.

### B. Step 2 - Supernet/Subnet Joint Train

In Step 2, $\boldsymbol{z_m}$ is rounded to an exact binary vector to define each $n_m$. We then jointly train the supernet with the selected subnets to further enhance performance. To improve training efficiency, in each training step, we apply the sandwich rules [15] by forwarding only three models at once: the smallest subnet, the largest subnet, and one medium subnet randomly sampled from the remaining $M-2$ subnets. To mitigate mutual interference between the supernet and subnets, we apply layer dropout [31] on layers where all groups from that layer are not selected for the smallest subnet. Additionally, we adapt the dropout magnitude in the feed-forward module (FFN) following [14], scaling it by $\frac{c_m}{c_s}$, where $c_s$, $c_m$ is the number of hidden channels in the supernet and subnet $n_m$ respectively.

### C. Selection Criteria and Granularity

We decompose the Conformer encoder into distinct parameter groups with varying granularities and assign each group a cost based on real-world requirements.
**Sparsity/FLOPs-aware Selection.** We compute the cost of each parameter group using two metrics: sparsity and the total number of floating-point operations (FLOPs). Sparsity, which is the proportion of zero-valued parameters in a model, directly affects model size and storage memory. FLOPs is indicative of inference cost, making it crucial for use cases where inference speed is critical. We use fvcore[1] to calculate the FLOPs.

---

[1] https://github.com/facebookresearch/fvcore

TABLE I: *ASR results comparison of different training methods, selection criteria, and granularities for training two models on the TED-LIUM-v2 test set.*

| Training method | Select criteria | Select granularity | Large WER [%] | Small Params. [M] | Small FLOPs [$10^8$] | Small WER [%] |
|---|---|---|---|---|---|---|
| separately | - | - | 7.8 | 21.6 | 2.53 | 8.2 |
| Aux-Loss | | block | 7.7 | | | 8.8 |
| *Simple-Top-k* | | layer | 7.7 | 20.7 | 2.47 | 8.3 |
| | sparsity | layer | 7.7 | 21.3 | 2.45 | 8.2 |
| | | component | 7.8 | 21.0 | 2.49 | 8.5 |
| | FLOPs | layer | 7.7 | 20.9 | 2.46 | 8.2 |
| | | component | 7.9 | 20.4 | 2.45 | 8.6 |
| $L_0$-Norm | sparsity | layer | 8.0 | 20.8 | 2.48 | 8.2 |
| | | component | 8.1 | 20.7 | 2.48 | 8.5 |
| *OrthoSoftmax* | - | layer | **7.6** | 22.8 | 2.62 | 8.3 |
| | sparsity | layer | 7.7 | 21.5 | 2.53 | 8.1 |
| | | component | 7.7 | 21.0 | 2.49 | 8.2 |
| | FLOPs | layer | 7.8 | 21.9 | 2.55 | 8.1 |
| | | component* | 7.7 | 21.5 | 2.52 | **8.0** |

* used as the baseline for Table II and Table III

**Layer/Component-wise Selection.** For layer-wise selection, we perform selection directly on entire FFN, convolution module (Conv), multi-headed self-attention module (MHSA) as in [22]. Inspired by [5], we extend the decomposition granularity to components: hidden channel chunks in FFN, attention heads in MHSA, and entire Conv layers.

## III. EXPERIMENTS

### A. Experimental Setup

Our experiments are conducted on 200h *TED-LIUM-v2* (*TED-v2*) [32] and 960h *Librispeech* (*LBS*) [33] dataset. We use a phoneme-based CTC model from the previous work [22] as baseline. The output labels are 79 end-of-word augmented phonemes [34]. The acoustic encoder consists of a VGG front end and 12 Conformer [25] blocks. We set the model size $d_{model}$ to 512 for *LBS* and 384 for *TED-v2* corpus, respectively. The number of attention heads is $\frac{d_{model}}{64}$ and FFN intermediate dimensions is $4 \times d_{model}$. For component-wise selection in Sec II-C, we set channel chunk size to $d_{model}$, so each FFN layer is decomposed into 4 chunks. The total number of parameters for *TED-v2* is 41.7M with $3.97 \times 10^8$ FLOPs, while for *LBS* it is 74.2M with $6.19 \times 10^8$ FLOPs. To ensure a fair comparison, we train for 730k steps in the *LBS* experiments and for 380k steps in the *TED-v2* experiments across all methods. The same learning rate schedule from [22] is used. We apply Viterbi decoding with a 4-gram word-level language model. We use RETURNN [35] to train the acoustic models and RASR [36] for recognition. All our config files and code to reproduce the results can be found online[2].

### B. 1 Supernet + 1 Subnet

Table I compares ASR results using different training methods, selection criteria, and granularities on a full-size supernet and a subnet about half its size. "Separately" refers to training and tuning the models individually for each size category. For the Aux-Loss, we add intermediate CTC loss

[2]https://github.com/rwth-i6/returnn-experiments/tree/master/2024-orthogonal-softmax

TABLE II: *WERs[%] for varying loss scales $\lambda$ and $\beta$ in training two models with OrthoSoftmax FLOPs-aware component-wise selection on the TED-LIUM-v2 dev and test set.*

| loss scale $\lambda$ | loss scale $\beta$ | WER [%] Large dev | Large test | Small dev | Small test |
|---|---|---|---|---|---|
| adaptive | 0.5 | 7.7 | 7.9 | **7.8** | 8.0 |
| | 1 | 7.7 | 7.8 | 7.9 | 8.2 |
| | 1.5 | 7.7 | 8.0 | 7.9 | 8.3 |
| | | **7.5** | **7.7** | 7.9 | 8.0 |
| 0.1 | $0 \to 1$ | **7.5** | 7.8 | 8.6 | 8.9 |
| 0.5 | | 7.6 | 7.9 | **7.8** | 8.3 |
| 1 | | 7.7 | 7.9 | **7.8** | 7.9 |

TABLE III: *ASR results of ablation study on TED-LIUM-v2 test and dev set. Baseline is * in Table I.*

| Ablation Study | WER [%] Large dev | Large test | Small dev | Small test |
|---|---|---|---|---|
| baseline | **7.5** | **7.7** | 7.9 | 8.0 |
| - softmax temperature annealing | 7.6 | **7.7** | **7.7** | **7.9** |
| - adaptive dropout in FFN | 7.6 | 7.9 | **7.7** | 8.1 |
| - layer dropout | **7.5** | **7.7** | 7.8 | 8.0 |

[37] to the output of 6-th Conformer block. We extend *Simple-Top-k* [22] by applying selection criteria. We also compare with $L_0$-Norm [38], which is widely used in pruning ASR models [5]–[8] to identify the subnet. We integrate $L_0$-Norm into Step 1 to calculate the mask $z$. We use the implementation from [39] and keep all other training aspects unchanged for a fair comparison. The results show that the WERs of *Simple-Top-k* and $L_0$-Norm degrade for both large and small models as select granularity becomes finer. This contrasts with the *OrthoSoftmax* results, where finer granularity improves subnet selection. Such behavior stems from inherent limitations in both methods. *Simple-Top-k* uses STE and suffers from gradient inconsistency. $L_0$-Norm uses a hard concrete distribution to relax the binary mask $z$ with a random variable $u \sim U(0, 1)$. While this makes $z$ differentiable, it also introduces training instability due to the fluctuating influence of $u$. Finer granularity increases binary elements in $z$ to be relaxed, exacerbating gradient approximation errors and hindering model convergence. In contrast, *OrthoSoftmax* updates $z$ smoothly across each training step.

### C. Ablation Study

Table II compares WERs across different loss scales. The term "adaptive" refers to using Eq.3, to dynamically adjust the subnet loss scale at each training step based on its CTC probability. The results show that compared to constant values, the adaptive approach effectively balances the supernet and subnet, resulting in strong performance for both. The term "$0 \to 1$" indicates that $\beta$ is linearly increased from 0 to 1 during Step 1. A larger $\beta$ worsens WER for smaller encoders, likely due to premature subnet selection from accelerated softmax orthogonalization. Contrarily, "$0 \to 1$" prioritizes ASR loss early, enabling more informed component selection later when their impact on ASR performance is clearer.

Table III shows the impact of each training method. Removing softmax $T$ annealing in Step 1 slightly improves WER for

TABLE IV: *ASR results of three encoders of large, medium, and small sizes on TED-LIUM-v2 test set*

| Training | Select criteria | Select granularity | Large WER [%] | Medium Parm. [M] | FLOPs [10^8] | WER [%] | Small Parm. [M] | FLOPs [10^8] | WER [%] |
|---|---|---|---|---|---|---|---|---|---|
| separately | - | - | 7.8 | 28.4 | 3.01 | 8.1 | 14.7 | 2.06 | 8.9 |
| Aux-Loss | | block | **7.6** | | | 8.0 | | | 10.1 |
| Ortho-Softmax | sparsity | layer | 7.7 | 27.7 | 2.96 | 8.0 | 14.9 | 2.06 | 9.0 |
| | | component | 7.8 | 28.1 | 2.99 | 8.1 | 14.8 | 2.06 | 8.8 |
| | FLOPs | layer | 7.7 | 28.3 | 3.00 | **7.9** | 15.3 | 2.09 | 8.7 |
| | | component | 7.8 | 28.3 | 3.00 | 8.0 | 15.0 | 2.07 | **8.6** |

TABLE V: *ASR results of three encoders of large, medium, and small sizes on Librispeech test-clean and test-other set*

| Training method | Select criteria | Select granl. | Large WER[%] cln | oth | Medium Prm. [M] | FLOPs [10^8] | WER[%] cln | oth | Small Prm. [M] | FLOPs [10^8] | WER[%] cln | oth |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| separately | - | - | 3.3 | 7.1 | 49.9 | 4.50 | 3.5 | 7.7 | 25.7 | 2.80 | **3.6** | **8.4** |
| Aux-Loss | | block | **3.2** | **6.9** | | | 3.6 | 7.9 | | | 4.5 | 9.7 |
| Ortho-Softmax | sparsity | layer | **3.2** | 7.1 | 49.8 | 4.48 | 3.3 | 7.4 | 25.0 | 2.74 | 3.8 | 9.0 |
| | | cmp | **3.2** | **6.9** | 49.8 | 4.48 | 3.3 | **7.3** | 24.4 | 2.71 | 3.7 | 8.8 |
| | FLOPs | layer | **3.2** | 7.1 | 48.7 | 4.41 | **3.2** | 7.4 | 26.1 | 2.81 | 3.7 | 8.7 |
| | | cmp | **3.2** | 7.0 | 49.8 | 4.48 | 3.4 | 7.5 | 25.6 | 2.79 | **3.6** | **8.4** |

small encoders. However, empirical observations show that for larger $k$, $T$ annealing can accelerate softmax orthogonalization and help convergence. Therefore, we retain it in our approach.

### D. 1 Supernet + 2 Subnets

Table IV and Table V report the results of jointly training three encoders on *TED-v2* and *LBS* test set. *OrthoSoftmax* significantly outperforms Aux-Loss on medium and small models. WER discrepancies increase as model size decreases, likely because Aux-Loss leverages the lower layers at the bottom, which are used to capture low-level features. For *OrthoSoftmax*, FLOPs-aware selection generally performs better than sparsity-aware selection. Using components as the granularity of selection proves more effective than using layers, especially for small-sized models. The WERs with FLOPs-aware component-wise selection are at least as good as those of the individually trained model across all model sizes and datasets, while significantly reducing training time.

### E. 1 Supernet + 4 Subnets

The results shown in Fig.1 are consistent with the observations in Sec III-D. For both *TED-v2* and *LBS* test sets, the red line closely matches the blue line, indicating that the FLOPs-aware component-wise approach achieves on-par WER across all five models compared to individual training.

### F. Remaining Ratio Analysis

Fig.2 shows that Convs are retained the most in all models, which is also observed in [5], [22], indicating that Convs are generally more important for the Conformer model. Besides, we observe that a large portion of the $1^{st}$ block is retained, even in the smallest model. This is likely because the $1^{st}$ block captures the initial temporal and phonetic characteristics. Also, fewer parameters are preserved in the lower layers compared to the higher layers, suggesting that the lower layers of the Conformer contain more redundancy. Furthermore, the MHSA heads are distributed sparsely across blocks. The authors of



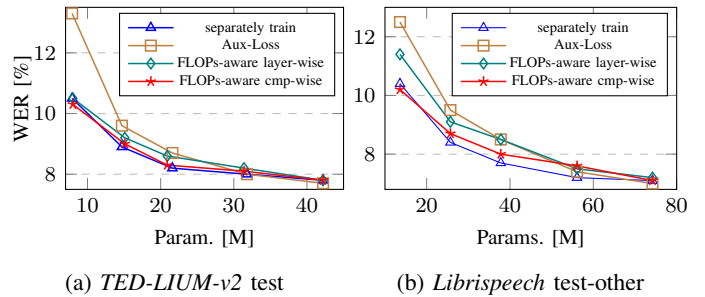(a) *TED-LIUM-v2* test     (b) *Librispeech* test-other

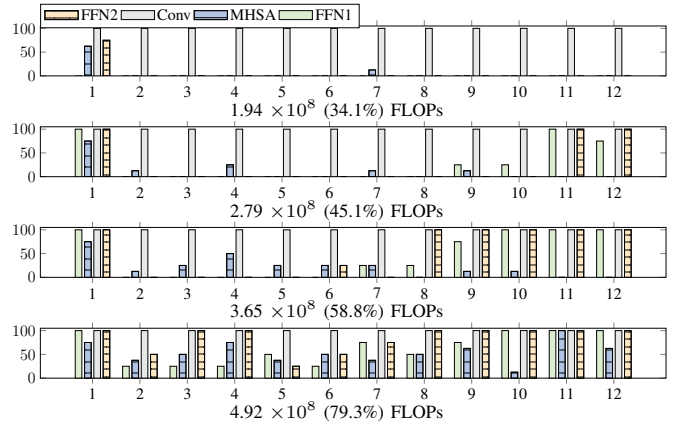Fig. 1: ASR results comparison of training five encoders.



Fig. 2: The remaining ratio for each layer in encoders of varying sizes, trained using the FLOPs-aware component-wise *OrthoSoftmax* method, as shown in Fig.1b.

[40] find that only a small subset of heads is crucial, and removing the vast majority of heads does not significantly affect performance. For smaller models, no MHSAs are preserved in the upper two layers, which aligns with the conclusion from [41] that these upper attention layers may be less useful due to the nearly diagonal attention score matrix. Although not shown in Fig.2, we observe that sparsity-aware approaches also select Convs the most. However, compared to the FLOPs-aware method, they favor FFN components over MHSA heads.

## IV. CONCLUSION

In this work, we introduce *OrthoSoftmax* for learning binary masks to identify optimal subnets during supernet training. Additionally, our approach enables subnet selection based on different levels of granularity and selection criteria. Experimental results show that FLOPs-aware component-wise selection performs overall the best. With the same number of training updates as a single training job, this approach achieves WERs comparable to or even better than those of individually trained models across various model sizes. By analyzing the selected components, we find that the $1^{st}$ Conformer block is important, and Convs are the most critical component.

## REFERENCES

[1] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," in *NeurIPS*, virtual, Dec. 2020.

[2] W. Hsu, B. Bolte, Y. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units," *IEEE ACM Trans. Audio Speech Lang. Process.*, vol. 29, pp. 3451–3460, 2021.

[3] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," in *ICML*, Hawaii, USA, July 2023, pp. 28492–28518.

[4] A. Gupta, T. Bau, J. Kim, Z. Zhu, S. Jha, and H. Garud, "Torque based Structured Pruning for Deep Neural Network," in *WACV*, Waikoloa, HI, USA, Jan. 2024, pp. 2699–2708.

[5] H. Jiang, L. L. Zhang, Y. Li, Y. Wu, S. Cao, T. Cao, Y. Yang, J. Li, M. Yang, and L. Qiu, "Accurate and Structured Pruning for Efficient Automatic Speech Recognition," in *Interspeech*, Dublin, Ireland, Aug. 2023, pp. 4104–4108.

[6] Y. Peng, K. Kim, F. Wu, P. Sridhar, and S. Watanabe, "Structured Pruning of Self-Supervised Pre-Trained Models for Speech Recognition and Understanding," in *ICASSP*, Greece, June 2023, pp. 1–5.

[7] Y. Peng, Y. Sudo, M. Shakeel, and S. Watanabe, "DPHuBERT: Joint Distillation and Pruning of Self-Supervised Speech Models," in *Interspeech*, Dublin, Ireland, Aug. 2023, pp. 62–66.

[8] H. Wang, S. Wang, W. Zhang, H. Suo, and Y. Wan, "Task-Agnostic Structured Pruning of Speech Representation Models," in *Interspeech*, Dublin, Ireland, Aug. 2023, pp. 4104–4108.

[9] H. Chang, S. Yang, and H. Lee, "Distilhubert: Speech Representation Learning by Layer-Wise Distillation of Hidden-Unit Bert," in *ICASSP*, Singapore, May 2022, pp. 7087–7091.

[10] Y. Lee, K. Jang, J. Goo, Y. Jung, and H. R. Kim, "FitHuBERT: Going Thinner and Deeper for Knowledge Distillation of Speech Self-Supervised Models," in *Interspeech*, Incheon, Korea, Sept. 2022, pp. 3588–3592.

[11] T. Ashihara, T. Moriya, K. Matsuura, and T. Tanaka, "Deep versus Wide: An Analysis of Student Architectures for Task-Agnostic Knowledge Distillation of Self-Supervised Speech Models," in *Interspeech*, Incheon, Korea, Sept. 2022, pp. 411–415.

[12] S. Ding, D. Qiu, D. Rim, Y. He, O. Rybakov, B. Li, R. Prabhavalkar, W. Wang, T. N. Sainath, Z. Han, J. Li, A. Yazdanbakhsh, and S. Agrawal, "USM-Lite: Quantization and Sparsity Aware Fine-Tuning for Speech Recognition with Universal Speech Models," in *ICASSP*, Seoul, Korea, Apr. 2024, pp. 10756–10760.

[13] O. Rybakov, P. Meadowlark, S. Ding, D. Qiu, J. Li, D. Rim, and Y. He, "2-bit Conformer Quantization for Automatic Speech Recognition," in *Interspeech*, Dublin, Ireland, Aug. 2023, pp. 4908–4912.

[14] Y. Shangguan, H. Yang, D. Li, C. Wu, Y. Fathullah, D. Wang, A. Dalmia, R. Krishnamoorthi, O. Kalinli, J. Jia, J. Mahadeokar, X. Lei, M. Seltzer, and V. Chandra, "TODM: Train Once Deploy Many Efficient Supernet-Based RNN-T Compression For On-device ASR Models," in *ICASSP*, Seoul, Korea, Apr. 2024, pp. 10216–10220.

[15] J. Yu, L. Yang, N. Xu, J. Yang, and T. S. Huang, "Slimmable Neural Networks," in *ICLR*, New Orleans, USA, May 2019.

[16] S. Ding, W. Wang, D. Zhao, T. N. Sainath, Y. He, R. David, R. Botros, X. Wang, R. Panigrahy, Q. Liang, D. Hwang, I. McGraw, R. Prabhavalkar, and T. Strohman, "A Unified Cascaded Encoder ASR Model for Dynamic Model Sizes," in *Interspeech*, Incheon, Korea, Sept. 2022, pp. 1706–1710.

[17] A. Narayanan, T. N. Sainath, R. Pang, J. Yu, C. Chiu, R. Prabhavalkar, E. Variani, and T. Strohman, "Cascaded Encoders for Unifying Streaming and Non-Streaming ASR," in *ICASSP*, Toronto, Canada, June 2021, pp. 5629–5633.

[18] Y. Shi, V. Nagaraja, C. Wu, J. Mahadeokar, D. Le, R. Prabhavalkar, A. Xiao, C. Yeh, J. Chan, C. Fuegen, O. Kalinli, and M. L. Seltzer, "Dynamic Encoder Transducer: A Flexible Solution for Trading Off Accuracy for Latency," in *Interspeech*, Brno, Czechia, Aug. 2021, pp. 2042–2046.

[19] H. Yang, S. Yuan, D. Wang, M. Li, P. Chuang, X. Zhang, G. Venkatesh, O. Kalinli, and V. Chandra, "Omni-Sparsity DNN: Fast Sparsity Optimization for On-Device Streaming E2E ASR Via Supernet," in *ICASSP*, Virtual and Singapore, May 2022, pp. 8197–8201.

[20] R. Wang, Q. Bai, J. Ao, L. Zhou, Z. Xiong, Z. Wei, Y. Zhang, T. Ko, and H. Li, "Lighthubert: Lightweight and Configurable Speech Representation Learning with Once-for-all Hidden-unit Bert," in *Interspeech*, Incheon, Korea, Sept. 2022, pp. 1686–1690.

[21] J. Lee, J. Kang, and S. Watanabe, "Layer Pruning on Demand with Intermediate CTC," in *Interspeech*, Brno, Czechia.

[22] J. Xu, W. Zhou, Z. Yang, E. Beck, and R. Schlüter, "Dynamic Encoder Size Based on Data-Driven Layer-wise Pruning for Speech Recognition," in *Interspeech*, Kos, Greece, Sept. 2024, To Appear.

[23] Y. Bengio, N. Leonard, and A. Courville, "Estimating or Propagating Gradients Through Stochastic Neurons for Conditional Computation," arXiv:1308.3432, Aug.2013.

[24] P. Yin, J. Lyu, S. Zhang, S. J. Osher, Y. Qi, and J. Xin, "Understanding Straight-Through Estimator in Training Activation Quantized Neural Nets," in *ICLR*, New Orleans, LA, USA, May 2019.

[25] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented Transformer for Speech Recognition," in *Interspeech*, Shanghai, China, Oct. 2020, pp. 5036–5040.

[26] A. Graves, S.Fernández, F.Gomez, and J.Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *ICML*, Pittsburgh, Pennsylvania, USA, June 2006, pp. 369–376.

[27] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in *ICCV*, Venice,Italy, Oct. 2017, pp. 2999–3007.

[28] W. Zhou, H. Wu, J. Xu, M. Zeineldeen, C. Lüscher, R. Schlüter, and H. Ney, "Enhancing and Adversarial: Improve ASR with Speaker Labels," in *ICASSP*, Rhodes Island, Greece, June 2023, pp. 1–5.

[29] A. Zhang, A. Chan, Y. Tay, J. Fu, S. Wang, S. Zhang, H. Shao, S. Yao, and R. Ka-Wei Lee, "On orthogonality constraints for transformers," in *ACL/IJCNLP*, Virtual, Aug. 2021, pp. 375–382.

[30] J. Wang, Y. Chen, R. Chakraborty, and S. X. Yu, "Orthogonal convolutional neural networks," in *CVPR*, Seattle, WA, USA, June 2020, pp. 11502–11512.

[31] A. Fan, E.Grave, and A. Joulin, "Reducing Transformer Depth on Demand with Structured Dropout," in *ICLR*, Addis Ababa, Ethiopia, Apr. 2020.

[32] A. Rousseau, P. Deléglise, and Y. Estève, "Enhancing the TED-LIUM Corpus with Selected Data for Language Modeling and More TED Talks," in *LREC*, Reykjavik, Iceland, May 2014, pp. 3935–3939.

[33] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR Corpus based on Public Domain Audio Books," in *ICASSP*, South Brisbane, Queensland, Australia, Apr. 2015, pp. 5206–5210.

[34] W. Zhou, S. Berger, R. Schlüter, and H. Ney, "Phoneme Based Neural Transducer for Large Vocabulary Speech Recognition," in *ICASSP*, Toronto, Canada, June 2021, pp. 5644–5648.

[35] A. Zeyer, T. Alkhouli, and H. Ney, "RETURNN as a Generic Flexible Neural Toolkit with Application to Translation and Speech Recognition," in *ACL*, Melbourne, Australia, July 2018, pp. 128–133.

[36] S. Wiesler, A. Richard, P. Golik, R. Schlüter, and H. Ney, "RASR/NN: the RWTH neural network toolkit for speech recognition," in *ICASSP*, Florence, Italy, May 2014, pp. 3281–3285.

[37] J. Lee, and S. Watanabe, "Intermediate Loss Regularization for CTC-Based Speech Recognition," in *ICASSP*, Toronto, Canada, Jun. 2021, pp. 6224–6228.

[38] C. Louizos, M. Welling, and D. P. Kingma, "Learning Sparse Neural Networks through $L_0$ Regularization," in *ICLR*, Canada, Apr. 2018.

[39] M. Xia, Z. Zhong, and D. Chen, "Structured Pruning Learns Compact and Accurate Models," in *ACL*, Ireland, May 2022, pp. 1513–1528.

[40] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *ACL*, Aug. 2019, pp. 5797–5808.

[41] S. Zhang, E. Loweimi, P. Bell, and S. Renals, "On the Usefulness of Self-Attention for Automatic Speech Recognition with Transformers," in *SLT*, Shenzhen,China, Jan. 2021, pp. 89–96.