Training dynamic models using early exits for automatic speech recognition on resource-constrained devices

George August Wright, Umberto Cappellazzo, Salah Zaiem, Desh Raj, Lucas Ondel Yang, Daniele Falavigna, Alessio Brutti

TRAINING DYNAMIC MODELS USING EARLY EXITS FOR AUTOMATIC SPEECH RECOGNITION ON RESOURCE-CONSTRAINED DEVICES

George August Wright[1], *Umberto Cappellazzo*[1], *Salah Zaiem*[2], *Desh Raj*[3], *Lucas Ondel Yang*[4], *Daniele Falavigna*[5], *Alessio Brutti*[5]

[1]University of Trento; [2]LTCI, Télécom Paris, Institut Polytechnique de Paris; [3]Johns Hopkins University; [4]Universite Paris-Saclay, LISN, CNRS; [5]Fondazione Bruno Kessler

ABSTRACT

The possibility of dynamically modifying the computational load of neural models at inference time is crucial for on-device processing, where computational power is limited and time-varying. Established approaches for neural model compression exist, but they provide architecturally static models. In this paper, we investigate the use of early-exit architectures, that rely on intermediate exit branches, applied to large-vocabulary speech recognition. This allows for the development of dynamic models that adjust their computational cost to the available resources and recognition performance. Unlike previous works, besides using pre-trained backbones we also train the model from scratch with an early-exit architecture. Experiments on public datasets show that early-exit architectures from scratch not only preserve performance levels when using fewer encoder layers, but also improve task accuracy as compared to using single-exit models or using pre-trained models. Additionally, we investigate an exit selection strategy based on posterior probabilities as an alternative to frame-based entropy.

Index Terms— dynamic models, early-exit, Conformer, ASR

1. INTRODUCTION

The edge-cloud continuum is an emerging complex ecosystem that integrates compute-enabled edge devices, distributing the overall computation workload among them [1]. Computational resources available on the devices considerably differ from each other and are time-varying due to sharing between different services. Therefore, having neural models that can dynamically change their trade-off between computation and performance is crucial. To this end, we investigate the use of early-exit architectures applied to large-vocabulary automatic speech recognition (ASR).

Previous work targeting neural models suitable for on-device processing mainly focused on decreasing the model size through compression [2], knowledge distillation [3, 4], pruning [5], and quantization [6]. Although very effective, these approaches deliver static solutions and require the models to be reconfigured each time the computational budget changes. Instead, it is preferable to dynamically adapt the model architecture to the memory and computational capabilities of each hosting device to avoid handling multiple models with varying trade-offs.

A solution for this task is represented by "early-exit" architectures that introduce intermediate exit branches [7, 8]. The input is not processed by all of the layers of the neural model but only by a subset of them, returning the result at an intermediate level and saving, in this way, the operations in the layers that are not traversed. An example is shown in Figure 1, where a layer-specific classifier/decoder

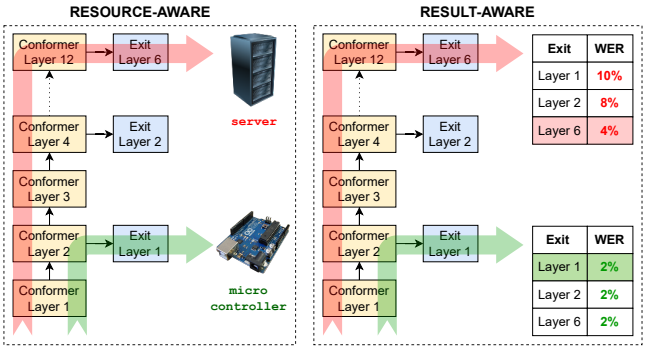

Fig. 1: Example of *resource-aware* and *result-aware* use of early-exits. On the left: the micro-controller can afford only two layers; the server can process the whole model. On the right: the first input requires processing the whole network; in the second case after 2 blocks the model already produces the best transcription.

(called "Exit Layer") is appended to some intermediate encoder layers. The motivation relies on the observation that, for easier inputs, the lower (earlier) layers have already learned a number of parameters sufficient to effectively predict the correct output. Early-exit architectures allow the development of **resource-aware** processing (Fig. 1, left) where the same model can be used on different devices, as well as **result-aware** processing (Fig. 1, right) where the model selects the earliest exit that would provide the same performance as processing the full network.

In this work, we investigate the use of early-exits applied to Conformer neural architectures, evaluated on three popular ASR benchmarks: LibriSpeech [9], TED-LIUM [10], and VoxPopuli [11]. While previous work on early-exit models for ASR mainly focused on inference exit selection using pre-trained large-scale models [12, 13, 14], we investigate training 3 different models both from scratch as well as pre-trained, using different early-exit losses as depicted in Fig. 2. We demonstrate that training the upstream network with the combined early-exit losses outperforms single-exit models that optimize only the loss of the final layer. Interestingly, early-exit training is found to be more effective when training the model from scratch, as opposed to fine-tuning an existing model. Overall, our contributions are:

1. We investigate early-exit training strategies for 3 different models, those trained from scratch as well as those initialized from pre-trained self-supervised models with different losses, showing that training the models from scratch is beneficial.
2. We compare early-exit selection methods based on entropy and confidence, and show that the N-best posterior provides a slightly

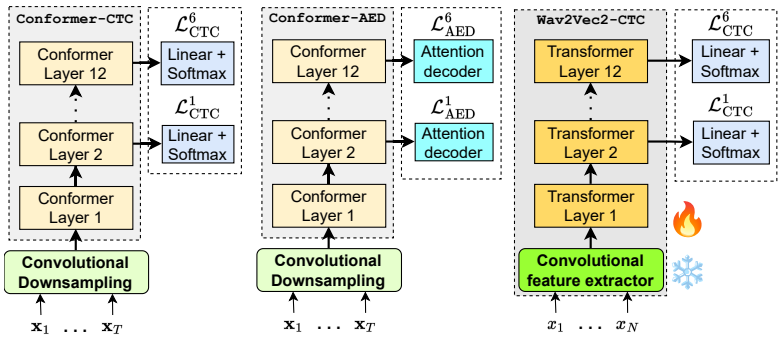

Fig. 2: Early-exit model architectures (from left to right): Conformer-CTC, Conformer-AED, and Wav2Vec2-CTC. Conformer-based models are entirely trained from scratch. Wav2Vec2-CTC is initialized with the pre-trained model and fine-tuned with early-exit losses, freezing the convolutional feature extractor.

Table 1: Hyperparameters for the early-exit model architectures shown in Fig. 2.

Feature	Conformer CTC	Conformer AED	Wav2Vec2 CTC
# params (M)	31.0	13.3	94.0
Encoder	12-layer Conf.	12-layer Conf.	12-layer Transf.
Attention dim.	256	144	768
Number heads	8	4	8
Feed-forward dim.	2048	1024	3072
Decoder	Linear	4-layer Transf.	Linear
Inputs	80-d MFCC	80-d MFCC	Waveform
Loss function	$\mathcal{L}_{\text{CTC}}$	$\mathcal{L}_{\text{CE}}$ & $\mathcal{L}_{\text{CTC}}$	$\mathcal{L}_{\text{CTC}}$
Output units	BPE	BPE	Grapheme
LM rescoring	✗	✓	✗
Data augmentation	✗	✓	✓

better trade-off than entropy.

3. To substantiate our claims, we perform experiments on 3 popular ASR benchmarks: LibriSpeech, TED-LIUM, and VoxPopuli.

2. RELATED WORK

"Early-exit" was introduced for computer vision in BranchyNet [7] by adding two branches to AlexNet [15]. The authors optimized the joint loss of the exits and also defined a confidence measure, based on the entropy of the output class distribution, to decide the exit level. More recently, Scardapane et al. [16] provided a theoretical framework for multi-exit neural architectures. Early classifiers have also been used on tiny (KB-sized) models [17]. Beside early-exit, other methods for dynamically selecting the model architecture for efficient inference (such as HydraNet [18]) have also been explored.

In speech recognition, early-exit was first introduced in HuBERT-EE [12] to speed up inference for a pre-trained HuBERT [19] model according to confidence measures, based on CTC confidence or output entropies, with no significant performance degradation. Similarly, Zaiem et al. [14] investigated different fine-tuning strategies in the context of a large pre-trained WavLM [20] model, comparing them with approaches based on layer removal and input downsampling.The overthinking issue of ASR encoders was also analysed in [21], where the authors reported theoretical lower bounds of speed/quality trade-offs for early-exit strategies. Exit selection strategies were proposed based on comparison between successive exits in terms of output distribution and transcriptions. Similar investigations using the entropy of the output distribution have also been conducted for recurrent neural networks [22].

All of the above investigations [12, 13, 14] employ pre-trained models by fine-tuning the transformer component, as is common for ASR. They are primarily *focused on efficient inference* by selecting the best early exit according to some criteria. An analogous observation can be made for natural language processing (NLP), where early-exit research has focused on accelerating inference of large pre-trained language models such as BERT [23, 24, 25]. Conversely, in this work, our objective is to *understand the training dynamics* of early-exit models (both trained from scratch and initialized from large pretrained models) by conducting exhaustive experiments on multiple datasets. We demonstrate that training the model from scratch, with joint optimisation of all exits, provides a significant performance improvements as compared to individual and pre-trained models (the latter, in particular, at the lowest exits).

3. EARLY-EXIT MODELS FOR ASR

Given an input sequence $\mathbf{X}$, such as a raw waveform $\{x_1, \ldots, x_N\}$ or acoustic features $\{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, where $\mathbf{x}_t \in \mathbb{R}^d$, an ASR system estimates the output sequence $\hat{\mathbf{y}}$ as

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{X}), \tag{1}$$

where $\mathbf{y} \in \mathcal{Y}^*$, for some vocabulary $\mathcal{Y}$, such as graphemes, phonemes, or BPE units. The distribution $P(\mathbf{y}|\mathbf{X})$ is usually estimated using a parameterized model Θ (such as a neural network), i.e., $P(\mathbf{y}|\mathbf{X};\Theta)$, which is learned using input-output pairs $(\mathbf{X},\mathbf{y})$.

For convenience, Θ is often factored into an encoder, which extracts high-dimensional representations $\mathbf{h}_1^T$ from $\mathbf{X}$, and a decoder, which maps $\mathbf{h}_1^T$ to the output sequence $\mathbf{y}_1^U$. Since $U \ll T$ in general, ASR decoders either use (i) an *alignment function* ($\mathcal{B} : \mathbf{a}_1^T \rightarrow \mathbf{y}_1^U$) for sequence training, or (ii) an *attention mechanism* with label-based cross-entropy training. We apply early-exit to ASR by adding decoders at several intermediate layers of the encoder (as shown in Fig. 2). Assuming that M such intermediate exits are added (with hypothesis $\hat{\mathbf{y}}^1, \ldots, \hat{\mathbf{y}}^M$), the overall model is trained by optimizing the joint objective

$$\mathcal{L}_{EE}(\hat{\mathbf{y}}^1, \ldots, \hat{\mathbf{y}}^M, \mathbf{y}) = \sum_{m=1}^{M} \mathcal{L}(\hat{\mathbf{y}}^m, \mathbf{y}), \tag{2}$$

where $\mathcal{L}(\hat{\mathbf{y}}^m, \mathbf{y}) = -\log P(\mathbf{y}|\mathbf{X};\Theta_m)$, and Θ_m denotes the subset of Θ used for exit m. In this work, we implement early-exit for several choices of the encoder and decoder, resulting in three models with different complexities, as described below. Hyperparameters for the models are summarized in Tab. 1.

Conformer-CTC: The Conformer encoder [26] is used to obtain $\mathbf{h}_1^T$, and the decoder is a linear layer with softmax. The intermediate loss function is connectionist temporal classification (CTC) [27], which is given as

$$\mathcal{L}_{\text{CTC}}(\hat{\mathbf{y}}, \mathbf{y}) = -\log \sum_{\mathbf{a}_1^T \in \mathcal{B}^{-1}(\mathbf{y}_1^U)} \prod_{t=1}^{T} P(a_t|\mathbf{h}_1^T), \tag{3}$$

where $a_t \in \mathcal{Y} \cup \{\phi\}$, and $\mathcal{B}$ maps $\mathbf{a}_1^T$ to $\mathbf{y}_1^U$ by removing repeated tokens and ϕ. We use a 12-layer encoder, and insert intermediate exits at all even-numbered layers.

Conformer-AED: To test the robustness of early-exits with complex decoders, we use an attention-based encoder-decoder (AED) model [28]. We retain the Conformer encoder as above, but replace the linear decoder with four transformer layers with cross-attention on $\mathbf{h}_1^T$. This decoder contains two output heads, trained with a CTC loss and a sequence-to-sequence cross-entropy loss respectively [29]. The overall loss function is given as

$$\mathcal{L}_{\text{AED}}(\hat{\mathbf{y}}, \mathbf{y}) = \lambda_{\text{CTC}}\mathcal{L}_{\text{CTC}}(\hat{\mathbf{y}}, \mathbf{y}) + \lambda_{\text{CE}}\mathcal{L}_{\text{CE}}(\hat{\mathbf{y}}, \mathbf{y}), \quad (4)$$

where $\mathcal{L}_{\text{CE}}(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_{u=1}^{U} \log P(y_u|\mathbf{h}_1^T, \mathbf{y}_1^{u-1})$, and λ's are hyperparameters. Following the SpeechBrain recipe [30], we set λ_{CTC} and λ_{CE} to 0.3 and 0.7, respectively. During inference, only the cross-entropy head is used , and a transformer-based language model trained with the same tokenization is used to rescore the hypothesis.

Wav2Vec2-CTC: Both the above models are trained *from scratch* by optimizing equation (Eq. 2). We also apply early-exit fine-tuning on a *pre-trained* Wav2Vec-2.0 [31] encoder using the joint CTC losses (Eq. 3). Unlike the above models, this model operates on raw waveforms processed using a convolutional feature extractor which is normally frozen during fine-tuning.

4. EARLY-EXIT SELECTION

To decide the exit for an early-exit model one can use a measure of its uncertainty, i.e., an exit layer is selected when its uncertainty drops below a given threshold which is, in turn, estimated to guarantee a desired performance level. Since the outputs from the encoder layers are converted to posterior probabilities by passing them to a softmax module, a suitable measure of their uncertainty is represented by their average frame entropies:

$$\Xi^m = -\frac{1}{T|\mathcal{Y}|}\sum_{t=1}^{t=T}\sum_{y\in\mathcal{Y}} P[y|\mathbf{h}_t^m]\log(P[y|\mathbf{h}_t^m]) \quad (5)$$

where, $P[y|\mathbf{h}_t^m]$ is the probability in the m^{th} encoder output at time t for each output token $y \in \mathcal{Y}$. While entropy is a common choice in literature, we also investigate a metric based on an estimate of the sentence confidence. This is computed by applying a softmax to the scores of the N-best hypotheses provided by each decoder:

$$\Psi^m = \frac{e^{s_1^m}}{\sum_1^K e^{s_k^m}} \quad (6)$$

where s_k^m is the log-probability of the k^{th} hypothesis at layer m, i.e. $s_k^m = \log(P[\hat{\mathbf{y}}_k^m|\mathbf{X}; \mathbf{\Theta}_m])$, and K is the number of N-best hypotheses. Preliminary experiments, aimed at finding the optimal performance/complexity trade-off, suggested the value $K = 300$.

5. EXPERIMENTS

We carry out experiments using LibriSpeech [9], TED-LIUM [10] and VoxPopuli [11]. LibriSpeech contains around 1,000 hours of read-aloud speech (audiobooks) partitioned into ≈960h to be used for training and ≈20h to be used for evaluation. TED-LIUM (release 3, [10]) comprises around 452 training hours of transcribed English speeches (from TED video conferences) and around 6 hours for evaluation. Finally, VoxPopuli is a multi-lingual corpus formed of around 400K hours of recordings (collected from European Parliament events). For this work, we used the English subset which consists of around 543 hours of training recordings and around 60 hours to be used for evaluation.

5.1. Implementation Details

As mentioned above, we consider 3 different models: Conformer-CTC (Eq. 3), Conformer-AED (Eq. 4), Wav2Vec2-CTC. The two Conformer models take as input 80 Mel Frequency Cepstral Coefficients (MFCCs) This MFCC sequence is passed through a series of 1D convolution sub-sampling layers. The output of this block is applied to a positional encoding module that feeds a stack of 12 Conformer blocks. The *Wav2Vec2.0* model (hereinafter referred to as *Wav2Vec-CTC*) also consists of a convolutional feature extractor followed by a 12-layer self-attention encoder, but takes as input raw waveforms.Both Conformer-CTC and Conformer-AED use a byte pair encoding (BPE) based tokenizer [32], with 256 and 5000 tokens respectively. Exit decoders of Wav2Vec2-CTC instead produces 32 grapheme-based tokens (28 characters + 1 blank token + 2 sentence boundary tokens + 1 unknown token) as per its official recipe.

The code for both training and inference for the Conformer-CTC and Wav2Vec2-CTC models is available[1], while the Conformer-CTC model is trained following the SpeechBrain recipe. Tab. 1 summarizes the main hyperparameters for the 3 models.

6. RESULTS

All results reported in this section are expressed in terms of word error rates (WERs) computed on the standard test partitions of the three datasets [2]. Tab. 2 reports the performance on LibriSpeech at different exits both training from scratch using the Conformer-CTC and Conformer-AED models, and fine-tuning Wav2Vec2-CTC. For each model we also report the performance of the corresponding single-exit model for comparison.

In our settings, the performance for the Conformer-CTC model with 12 layers is 6.6% on test-clean and 17.7% on test-other. As expected, the WER is higher in the lower layers. The performance significantly decreases only in the lowest two exits (exits 2 and 4), while it remains not far from the best one (that of layer 12) in the middle layers (i.e., 6, 8 and 10), which, however, require significantly fewer parameters. Similar trends are achieved with the Conformer-AED model but with significantly better absolute performance (2.3% and 6.0% WER in the highest layer for test-clean and test-other, respectively). This absolute improvement is attributed both to the use of transformer-based decoders as well as to the language model rescoring, allowing the model to reach state-of-the-art on LibriSpeech. Tab. 2 shows that the Wav2Vec2-CTC model exhibits a behaviour similar to the two Conformer models. However, since the pre-trained Wav2Vec2-CTC model has been optimised solely on the loss of the highest layer, it leads to very high WERs at the lower exits with an evident performance gain already at the 8th layer. This degradation is much less evident in the Conformer models trained from scratch.

In summary, although smaller and trained on less data, the Conformer-CTC/AED models perform better than Wav2Vec2-CTC in the last 3 layers. **These results suggest that for early-exit architectures, training a model from scratch is more efficient than fine-tuning a large and accurate model not pre-trained with early-exits.** It is worth noting that the same trends are observed considering different decoders, different training losses, and independently of the use of a language model.

Another important outcome emerges as we compare the performance achieved with the early-exit models with those obtained

[1] https://github.com/augustgw/early-exit-transformer and https://github.com/augustgw/wav2vec2-ee

[2] Results on the dev-sets are not reported for the sake of space but are available.

Table 2: WERs on the **LibriSpeech** at different exits, obtained with the 3 models under investigation. "layer" indicates at which layer the exit is located or the number of layers of the single-exit model. EE indicates that the model has been trained with the early-exit losses while no-EE refers to the single-exit model.

Layer	Conformer-CTC				Conformer-AED				Wav2Vec2-CTC			
	test-clean		test-other		test-clean		test-other		test-clean		test-other	
	no-EE	EE	no-EE	EE	no-EE	EE	no-EE	EE	no-EE	EE	no-EE	EE
2	17.6	23.9	36.1	43.8	18.9	20.1	38.0	40.1	35.7	33.7	56.7	56.0
4	9.8	11.6	24.3	25.7	12.8	12.5	25.8	25.2	17.4	17.4	35.5	36.7
6	7.6	6.8	20.0	18.1	8.4	7.7	20.1	17.1	10.7	9.6	24.8	23.7
8	–	5.9	–	16.3	–	4.4	–	11.5	–	5.8	–	15.9
10	–	5.2	–	15.8	–	2.8	–	6.9	–	4.5	–	12.6
12	6.5	5.1	17.7	15.1	2.5	2.3	6.1	6.0	3.4	4.3	8.6	12.2

Table 3: WERs on the **TED-LIUM** and **VoxPopuli** at different exits, training the model Conformer-CTC from scratch.

Layer	TED-LIUM		VoxPopuli	
	no-EE	EE	no-EE	EE
2	42.7	43.8	27.3	36.7
4	35.4	23.4	19.7	21.1
6	25.5	18.0	18.7	17.3
8	–	16.1	–	15.4
10	–	14.9	–	14.7
12	16.4	14.6	16.3	14.3

with the corresponding single-exit models (column "no-EE" in Tab. 2). Apart from the lowest exits (layers 2 and 4), the single-exit Conformer-CTC/AED models deliver worse WERs than the early-exit counterpart. This indicates the beneficial effects of the compound loss, acting as a regularizer and improving both robustness and generalization. This observation is in line with previous studies applying losses at lower layers, although with different granularity than an ASR decoder [16, 7, 33]. In other words, using a single model with multiple exits not only reduces the computational burden of training multiple single exit models but also delivers better performance. Note that this claim is not valid for the top layer of the Wav2Vec2-CTC model where the performance decreases (from 3.4% to 4.3% on test-clean and from 8.6% to 12.2% on test-other) when fine-tuning with the early-exit loss. In this case, however, it has to be considered that the model has not been trained from scratch but, as usual, its convolutional feature encoder has been frozen and only the transformer encoder module has been fine-tuned.

Finally, experiments on TED-LIUM and VoxPopuli, shown in Tab. 3, confirm the observations drawn on LibriSpeech. In these experiments, we also observe superior performance in models trained with the compound early-exit loss as compared to those trained with single exits, for layers higher than 4.

6.1. Exit selection during inference

Having assessed the efficacy of early-exit architectures for resource-aware processing (i.e., for each individual exit), we now analyse the behavior of the different models when selecting the exit, using either the average frame entropy (Eq. 5) or the sentence confidence (Eq. 6), i.e., addressing a result-aware solution. In both cases, we follow the common practice, implementing a thresholding approach: given a predefined threshold, we select the first exit whose entropy is below that value or whose posterior is above. Previous studies [14, 13] have observed that although the overall performance of lower layers is inferior to those processing the whole network (i.e., the final layers), in many cases the performance is on par. Being able to identify those cases would considerably reduce the overall computational cost. Fig. 3 shows the average exit (y-axis) with the corresponding WER (x-axis) when varying the selection threshold for the three models and the two metrics. The closer the curve to the chart origin, the better. We observe that, as expected, better models deliver better performance in exit selection: the Conformer-AED lines are well below the others. Sentence confidence (dotted lines) on average selects lower exits than entropy at the same WER values.

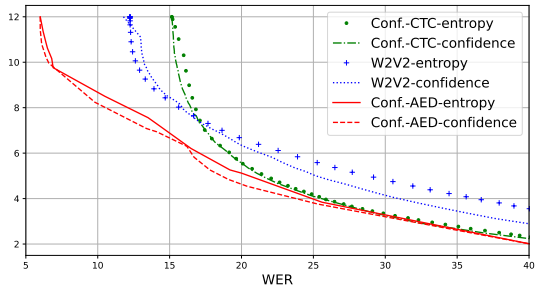

Fig. 3: Average-exit selection and WER varying the exit selection threshold for the 3 models and using both entropies and sentence confidence as exit metrics.

7. CONCLUSION AND FUTURE WORKS

In this paper, we investigated early-exit architectures for ASR by comparing the training and inference of three models, two based on a Conformer architecture and one based on Wav2Vec2. We demonstrated the benefits of training models from scratch using early-exit, as compared to fine-tuning a pre-trained model, on three datasets. Future works will investigate weighting schemes for the compound loss in Eq. 2 or alternative training strategies [16], including distillation (similar to [33]) from upper layers of the model.

Acknowledgments. This work was partially funded by the PNRR ICSC National Research Centre for High Performance Computing, Big Data and Quantum Computing (CN00000013), under the NRRP MUR program funded by the NextGenerationEU. We also acknowledge support from the JSALT 2023 workshop, hosted at Le Mans University, France, and sponsored by Johns Hopkins University with unrestricted gifts from Amazon, Facebook, Google, and Microsoft.

8. REFERENCES

[1] K. Liang, R. Ma, Y. Hua, H. Wang, N. Hu, T. Song, H. Gao, and H. Guan, "Wh2d2n2: Distributed ai-enabled ok-asn service for web of things," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 22, pp. 1 – 16, 2023.

[2] G. Cerutti, R. Prasad, A. Brutti, and E. Farella, "Compact recurrent neural networks for acoustic event detection on low-energy low-complexity platforms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 4, pp. 654–664, 2020.

[3] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.

[4] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "KL-Divergence Regularized Deep Neural Network Adaptation for Improved Large Vocabulary Speech Recognition," in *Proc. of ICASSP*, Vancouver (Canada), May, 26-31 2013, pp. 7893–7897.

[5] H. Song, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," in *Proc. of 4th International Conference on Learning Representations (ICLR)*, 2016.

[6] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proceeding of CVPR*, 2016, pp. 4820–4828.

[7] T. Teerapittayanon, B. McDanel, , and H. Kung, "BranchyNet: Fast Inference via Early Exiting from Deep Neural Networks," *arXiv:1709.01686*, 2017.

[8] M. Phuong and Lampert, "Distillation-based training for multi-exit architectures," in *Proc. of ICCV*, 2019.

[9] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. of ICASSP*, 2015, pp. 5206–5210.

[10] F. Hernandez, V. Nguyen, S. Ghannay, N. Tomashenko, and Y. Estève, "TED-LIUM 3: Twice as Much Data and Corpus Repartition for Experiments on Speaker Adaptation," in *Speech and Computer*, A. Karpov, O. Jokisch, and R. Potapova, Eds. 2018, pp. 198–208, Springer International Publishing.

[11] C. Wang, M. Riviere, A. Lee, A. Wu, C. Talnikar, D. Haziza, M. Williamson, J. Pino, and E. Dupoux, "VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation," in *Proceedings of the ACL*. Aug. 2021, Association for Computational Linguistics.

[12] J. W. Yoon, J. W. Beom, and N. S. K., "HuBERT-EE: Early Exiting HuBERT for Efficient Speech Recognition," *arXiv:2204.06328*, 2022.

[13] D. Berrebbi, B. Yan, and S. Watanabe, "Avoid overthinking in self-supervised models for speech recognition," in *IEEE ICASSP*, 2022.

[14] S. Zaiem, R. Algayres, T. Parcollet, S. Essid, and M. Ravanelli, "Fine-tuning strategies for faster inference using self-supervised models: a comparative study," *arXiv:2303.06740*, 2023.

[15] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[16] S. Scardapane, M. Scarpiniti, E. Maccarelli, and M. Uncini, "Why should we add early exits to neural networks?," *Cognitive Computation*, vol. 12, no. 5, pp. 954–966, 2020.

[17] N. P. Ghanathe and S. Wilton, "T-recx: Tiny-resource efficient convolutional neural networks with early-exit," in *Proceedings of ACM International Conference on Computing Frontiers*, 2023, pp. 123–133.

[18] R. T. Mullapudi, W. R. Mark, N. M. Shazeer, and K. Fatahalian, "Hydranets: Specialized dynamic architectures for efficient inference," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8080–8089, 2018.

[19] W. N. Hsu, B. Bolte, Y. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3451–3460, 2021.

[20] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao, J. Wu, L. Zhou, S. Ren, Y. Qian, Y. Qian, M. Zeng, and F. Wei, "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.

[21] D. Berrebbi, B. Yan, and S. Watanabe, "Avoid overthinking in self-supervised mmodels for speech recognition," *arXiv:2211.08989*, 2022.

[22] R. Tang, K. V. S. M. Kumar, J. Xin, P. Vyas, W. Li, G. Yang, Y. Mao, C. Murray, and J. Lin, "Temporal early exiting for streaming speech commands recognition," in *Proc. of ICASSP*, 2022.

[23] W. Zhou, C. Xu, T. Ge, J. McAuley, K. Xu, and F. Wei, "Bert loses patience: Fast and robust inference with early exit," in *NIPS*, 2020.

[24] W. Liu, P. Zhou, Z. Zhao, Z. Wang, H. Deng, and Q. Ju, "FastBERT: a self-distilling BERT with adaptive inference time," in *ACL*, 2020.

[25] J. Xin, R. Tang, J. Lee, Y. Yu, and J. Lin, "DeeBERT: Dynamic early exiting for accelerating bert inference," in *ACL*, 2020.

[26] Y. Lu, Z. Li, D. He, Z. Sun, B. Dong, T. Qin, L. Wang, and T. Liu, "Understanding and improving transformer from a multi-particle dynamic system point of view," *arXiv:1906.02762*, 2019.

[27] A. Graves, S. Fernández, F. J. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[28] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *NIPS*, 2015.

[29] S. Kim, T. Hori, and S. Watanabe, "Joint ctc-attention based end-to-end speech recognition using multi-task learning," in *IEEE ICASSP*, 2016.

[30] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio, "SpeechBrain: A general-purpose speech toolkit," 2021, arXiv:2106.04624.

[31] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," 2020.

[32] S. R., B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. of ACL*, 2016, pp. 1715—1725.

[33] S. Geng, P. Gao, Z. Fu, and Y. Zhang, "Romebert: Robust training of multi-exit bert," *arXiv, preprint arXiv:2101.09755*, 2021.