DISTILLATION-BASED LAYER DROPPING (DLD): EFFECTIVE END-TO-END FRAMEWORK FOR DYNAMIC SPEECH NETWORKS

Abdul Hannan[1,2]*, Daniele Falavigna*[2]*, Shah Nawaz*[3]*, Mubashir Noman*[4]*, Markus Schedl*[3,5]*, Alessio Brutti*[2]

[1]University of Trento, Italy, [2]Fondazione Bruno Kessler, Italy,
[3]Johannes Kepler University Linz, Austria, [4]MBZUAI U.A.E.,
[5]Linz Institute of Technology, Austria

ABSTRACT

Edge devices operate in constrained and varying resource settings, requiring dynamic architectures that can adapt to limitations of the available resources. To meet such demands, layer dropping ($\mathcal{LD}$) approach is typically used to transform static models into dynamic ones by skipping parts of the network along with reducing overall computational complexity. However, existing $\mathcal{LD}$ methods greatly impact the dynamic model's performance for low and high dropping cases, deteriorating the performance-computation trade-off. To this end, we propose a distillation-based layer dropping (DLD) framework that effectively combines the capabilities of knowledge distillation and $\mathcal{LD}$ in an end-to-end fashion, thereby achieving state-of-the-art performance for dynamic speech networks. Comprehensive experimentation utilizing well-known speech recognition methods, including conformer and WavLM, on three public benchmarks demonstrates the effectiveness of our framework, reducing the word error rate by 9.32% and 2.25% for high and no dropping cases with 33.3% reduction in training time.

Index Terms— Automatic speech recognition, dynamic model, feature alignment, knowledge distillation, layer drop, layer skip

1. INTRODUCTION

Speech foundation models offer promising performance on various downstream tasks such as automatic speech recognition (ASR), spoken language understanding, speaker identification etc., due to their rich semantic representation capability. This favorable performance is achieved at the cost of enormous computational complexity limiting their deployment on the resource-constrained devices with resource variability. Conventional techniques like pruning [1, 2], quantization [3] and knowledge distillation [4] renders compression with relatively good performance-computation trade-off, however, these methods do not suffice the requirements of a dynamic architecture that is preferred for varying-resource devices. To achieve dynamic architectures that are well-suited for such scenarios, techniques like *data-offloading:* presents privacy and latency issues [5, 6], *early exit:* involves delicate design choices for inserting exit branches [7], and especially *layer dropping (or skipping)* are utilized enabling the architecture to perform efficiently in such dynamic environment.

Layer dropping ($\mathcal{LD}$), inspired from stochastic depth [8] in vision domain, has been increasingly utilized to transform static architectures into dynamic ones [8, 9, 10]. $\mathcal{LD}$, also referred as structured pruning, executes some parts of the network and skips the rest, providing regularization during training phase and improving execution time, and can be categorized as: (i) *random dropping* (RD): where each module can be dropped depending on probability [9, 11, 12, 13], and (ii): *data-driven*: where input of the model constitutes the dropping blueprint [10, 14, 15, 16, 17]. Zhang and He [11] proposed a gating scheme for RD to skip different subnets of pre-trained networks using BERT, RoBERTa, and XLNet architectures. In context of ASR domain, recent work [18] presented an effective two-step framework to create multiple small sized static models, however, the training recipe requires enhancement as well as the resulting models are not suitable for varying resource environments. Another study [12] compared the performance of different approaches, i.e., early exit, input downsampling, and RD with a dropping probability $p_d = 0.5$, by fine-tuning a pre-trained network on a subset of dataset. However, they omit the evaluation for extreme dropping scenarios. LDASR [13] investigated the inference time behavior of conformer architecture trained using RD with $p_d = \{0.2, 0.5, 0.8\}$, and revealed that $p_d = 0.5$ provides optimum performance-computation trade-off. However, their approach offers substantial performance degradation for high dropping values as well as struggles to exploit the computational capacity of the complete model in case of no (or low) dropping. Similarly, other works attempts to achieve a dynamic network using learnable masking [17], and data-driven methods [15, 19], and either omit evaluation or offer significant performance drop for high dropping values.

Besides, knowledge distillation (KD) has been extensively utilized to achieve model compression for ASR downstream task, resulting in compressed static models. TutorNet [20] performs KD from multiple locations of teacher model,

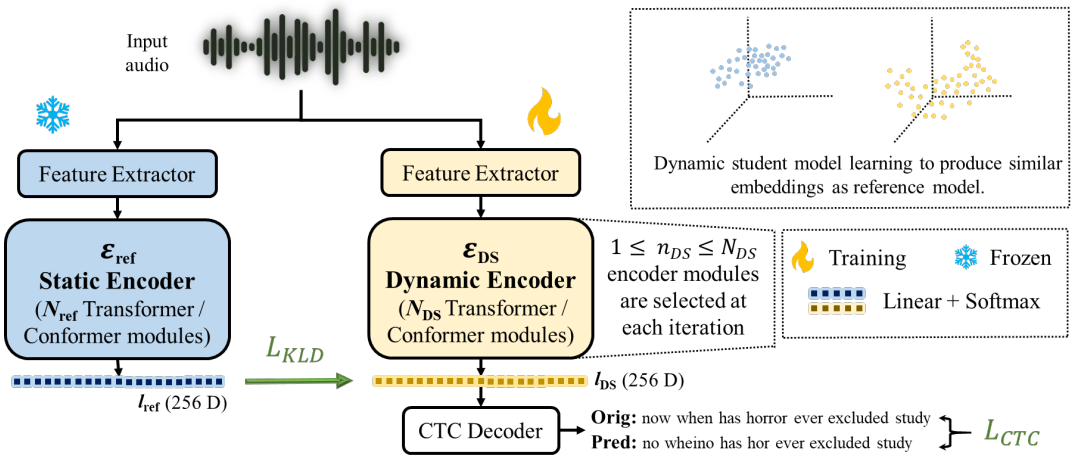

Fig. 1. Illustration of proposed DLD framework. $\mathcal{M}_{\text{ref}}$ uses all encoder layers to supervise the embeddings of $\mathcal{M}_{\text{DS}}$.

whereas others [21, 22] utilized several teacher networks to supervise the student model. In this regard, inspired from two-step framework of [18], this work aims to address the aforementioned issues of dynamic architectures using knowledge distillation in conjunction with random dropping and is the first of its kind to employ KD with dynamic architectures for ASR to the best of our knowledge.

To summarize, our contributions are:

1. We propose an end-to-end DLD framework that distills expert knowledge of large model to the dynamic student model, featuring promising performance.
2. The introduced approach resolves the performance degradation issue for no dropping, in addition to improving overall performance for all model sizes.
3. We perform comprehensive experimentation on two publicly available ASR benchmarks using conformer and WavLM architectures, demonstrating the superiority of proposed framework.

2. METHODOLOGY

To enhance the inference time efficacy of a dynamic model and mitigate performance degradation issues, we propose an end-to-end DLD[1] framework that aligns the distribution of latent embeddings of a dynamic student model $\mathcal{M}_{\text{DS}}$ to a reference model $\mathcal{M}_{\text{ref}}$.

Architecture and Data Flow: $\mathcal{M}_{\text{ref}}$ consists of a feature extractor $\mathcal{F}_{\text{ref}}$, encoder $\mathcal{E}$ with N_{ref} transformer/conformer modules, and a linear projector whose output undergoes a softmax operation to get latent embedding l_{ref}. Similarly, $\mathcal{M}_{\text{DS}}$ comprises of a feature extractor $\mathcal{F}_{\text{DS}}$, encoder $\mathcal{E}_{\text{DS}}$ with a total of N_{DS} transformer/conformer modules from which n_{DS} are used per iteration, and a linear projector followed by a softmax, providing latent embedding l_{DS}. The $\mathcal{E}_{\text{DS}}$ is equipped with a gated mechanism that allows to process or skip the i-th encoder module using a gate g^i. The gate $g^i \in \{0, 1\}$ follows the bernoulli distribution (BD) with a probability of 0.5.

$$y^{i+1} = y^i + g^i \cdot f^i(y^i) \tag{1}$$

where y^i and y^{i+1} are the input and output of i-th encoder block, and $f^i(.)$ is the transformation applied by the i-th encoder. As g^i follows the bernoulli distribution, n_{DS} also becomes a bernoulli random variable whose value can be determined as $n_{\text{DS}} = \sum_{i=1}^{N_{\text{DS}}} g^i$ for each iteration during training period. At inference, the value of n_{DS} is defined enabling us to evaluate the model's performance to different encoder depths, i.e. $n_{\text{DS}} = 2, 4, \ldots, N_{\text{DS}}$.

For a dataset $\mathcal{D} \in \{(a^j, t^j)\}_{j=1}^{J}$ comprising of J audio-transcription pairs, $\mathcal{M}_{\text{ref}}$ produces a latent space representation $l_{ref}^j|\mathcal{E}_{N_{\text{ref}}}$ for the j-th input using N_{ref} encoder modules. In contrast, $\mathcal{M}_{\text{DS}}$ utilizes n_{DS} number of encoder modules for each iteration to produce the latent space representation $l_{\text{DS}}^j|\mathcal{E}_{n_{\text{DS}}}$ where $1 \leq n_{\text{DS}} \leq N_{\text{DS}}$.

Objective Function: To align the distributions of reference model's embedding $l_{\text{ref}}^j|\mathcal{E}_{N_{\text{ref}}}$ and dynamic student model's embedding $l_{\text{DS}}^j|\mathcal{E}_{n_{\text{DS}}}$, we employ Kullback–Leibler divergence ($\mathcal{L}_{\text{KLD}}$) loss which minimizes the statistical distance between both embeddings.

$$\mathcal{L}_{\text{KLD}} = \min \sum_{j=i}^{J} \mathrm{D}_{\text{KL}}\left(l_{\text{ref}}^j \,||\, l_{\text{DS}}^j\right) \tag{2}$$

In addition, the token probabilities $l_{\text{DS}}^j|\mathcal{E}_{n_{\text{DS}}}$ produced by $\mathcal{M}_{\text{DS}}$ for input a^j, are forwarded to a Connectionist Temporal Classification (CTC) decoder [23] to generate predicted text, which is employed to estimate CTC loss ($\mathcal{L}_{\text{CTC}}$) by comparing with ground-truth transcription t^j.

$$\mathcal{L}_{\text{CTC}} = \min \sum_{j=1}^{J} \mathbf{F}_{\text{CTC}}\left(l_{\text{DS}}^j|\mathcal{E}_{n_{\text{DS}}}, t^j\right) \tag{3}$$

The overall loss function to be minimized is given as:

$$\mathcal{L} = \mathcal{L}_{\text{KLD}} + \mathcal{L}_{\text{CTC}} \tag{4}$$

[1]Our code is available online https://github.com/hannabdul/DLD4ASR

Table 1. Comparing WER (in %) of proposed framework on conformer architecture against RD based baseline methods when trained on LibriSpeech 1000. RD - random dropping, Params column depicts number of executed parameters.

n_{DS}	LibriSpeech			TEDLIUM v3			Params
	RD_{sc}	RD_{LD}	Ours	RD_{sc}	RD_{LD}	Ours	M
12	8.07	7.69	5.82	13.72	14.03	12.36	31.2
10	7.28	6.95	5.90	14.01	13.94	12.69	26.06
8	7.28	6.92	6.32	15.12	14.25	13.05	20.91
6	8.39	7.98	7.32	17.24	15.78	14.89	15.76
4	12.81	12.54	11.81	24.46	22.43	20.88	10.61
2	39.87	43.62	38.30	54.20	55.30	51.40	5.47
$\mathcal{M}_{\text{ref}}$	5.29			11.83			31.2

3. EXPERIMENTATION AND RESULTS

Architectures: We evaluated the proposed DLD framework using two architectures: (i) Conformer: a modified version with light feature extractor along with linear projector instead of LSTM-based projector (similar to [13, 16]), (ii) WavLM-base model [24]. For conformer model, mel-spectrograms are extracted for each 320 samples (20 ms) of input sequence with a hop-length of 160 samples (10 ms), whereas the input sequence is fed directly to the WavLM model.

Implementation Details: For both architectures, we kept the reference model frozen and finetuned the dynamic student model using CTC loss on a NVIDIA $A40$ GPU. In case of Conformer, the weights of $\mathcal{M}_{\text{DS}}$ are initialized with pre-trained weights of static model and finetuned for 100 epochs using batch size of 64 and L2 regularization of $5e^{-4}$. The learning rate is increased for 10k warm-up steps and exponentially decreased till the end of training. In case of WavLM-base, we initialized $\mathcal{M}_{\text{DS}}$ with pre-trained weights and finetuned using Adam optimizer with a batch size of 8 and default configuration settings. We measured the architecture's performance in terms of Word Error Rate (WER).

Datasets: For ASR downstream task, we employed two publicly available corporas: LibriSpeech-1000 [25] and TED-LIUM v3 [26].

Baselines: For conformer architecture, we compare against Random Dropping (RD) method with dropping probability $p_d = 0.5$ [13] (referred as "RD_{LD}" in this paper) that is trained for $300+$ epochs and an input-driven dropping strategy [15] (referred as "I3D") trained on 100 hours of librispeech corpora. Additionally, we compare against a conformer trained from scratch "RD_{sc}" with $p_d = 0.5$ for 150 epochs. For WavLM, we compared against RD with $p_d = 0.5$ [16] (referred as "RD_{w-sc}" in this paper). Unless stated, all training procedures are performed using complete datasets $\big($ librispeech (1000hrs) and tedlium-v3 (452hrs) $\big)$.

Table 2. Comparing measured WER (in %) for finetuning WavLM with RD with and without DLD framework.

n_{DS}	LibriSpeech		TEDLIUM v3		Params	Speed-up
	RD_{w-sc}	Ours	RD_{w-sc}	Ours	M	
12	5.47	4.57	10.32	9.19	94.40	1x
10	5.78	4.66	10.75	9.24	80.22	1.17x
8	6.52	5.20	12.53	10.55	66.04	1.43x
6	9.01	6.73	17.74	13.89	51.87	1.82x
4	17.59	12.76	31.90	24.61	37.69	2.50x
2	60.10	50.78	76.76	68.79	23.51	4.01x
$\mathcal{M}_{\text{ref}}$	3.5		10.12		94.40	1x

3.1. Results

3.1.1. Conformer

Table 1 and 2 enlists the measured WER using aforementioned architectures on LibriSpeech test-clean and TED-LIUM v3 test sets. For conformer architecture, it is evident from Table 1 that our framework, leveraging from the knowledge distillation to train the dynamic student model, outperforms models trained without such knowledge transfer. We notice that RD_{sc} and RD_{LD} models are capable of adapting to varying resource settings, however, their performance degrades when evaluated with less or no dropping ($n_{\text{DS}} \geq 8$), which is superbly resolved by proposed DLD. We observe a trend of improved performance-computation trade-off on test splits of both datasets, in addition to retaining performance with low or no inference time dropping (2.25% and 1.67% less WER for $n_{\text{DS}} = 12$ without dropping any encoder module from respective baseline models). Moreover, for $n_{\text{DS}} = 6$, our framework achieves 2.35% and 0.89% improved WER with 2x computational speed-up on TED-LIUM v3 test split. Similarly, when trained on librispeech-100 split, our method outclass I3D (input-driven strategy) [15] that contains transformer-based encoder structure with deep feature extractor and triple number of encoder layers. They report a WER of $\approx$13.5% and $\approx$12.2% for dropping 50% and 25% of encoder modules for librispeech test-clean split, which is $\approx 3.56\%$ and $\approx 4.06\%$ higher than the proposed framework.

3.1.2. WavLM

To transform a static WavLM foundation model into a dynamic one, our framework significantly improves the performance-computation trade-off as illustrated in Table 2. For instance, we achieve 4.83% and 7.29% better WER on LibriSpeech and TEDLIUM datasets than the baseline with random dropping. Furthermore, our framework surpasses learnable masking method proposed in [17] by 0.37%, 4.57%, and 10.65% for dropping of 25%, 50%, and 75% encoder modules. We observe from Figure 2 that [17] also

Table 3. Evaluation of proposed framework on LibriSpeech test-clean split using Conformer model as training progresses.

$n_{\mathbf{DS}}$	RD_{sc}	RD_{LD}	WER at Epoch			
			25	**50**	**75**	**100**
12	8.07	7.69	6.39	6.09	5.89	5.82
10	7.28	6.95	6.47	6.19	6.04	5.90
8	7.28	6.92	6.93	6.64	6.40	6.32
6	8.39	7.98	8.16	7.74	7.49	7.32
4	12.81	12.54	13.26	12.58	12.09	11.81
2	39.87	43.62	41.26	39.77	38.82	38.30

provides reduction in model size, however, they employed Wave2Vec2 foundation model containing $317M$ parameters (3.37x more than ours) along with a bi-LSTM layer (linear projection in our case). The number of utilized parameters of their 75% trimmed model is similar to our full-sized model with substantially low performance (33.9% vs 4.57%). Figure 2 illustrates the variation in computational load and performance-computation trade-off between WavLM (ours) and wav2vec2 [17].

On performance degradation : Table 1 highlights that DLD framework resolves the inference time degradation problem present in the conformer baseline methods (RD_{sc} and RD_{LD} [13]). For $n_{\text{DS}} = \{12, 10, 8\}$, our framework scores WER of $\{5.82, 5.90, 6.32\}$ with minimal performance drop for no and low dropping values, whereas RD_{sc} and RD_{LD} gives WER of $\{8.07, 7.28, 7.28\}$ and $\{7.69, 6.95, 6.92\}$ respectively, depicting substantial performance degradation. We attribute this performance limitation of baseline methods to the shallow feature extractor which fails to provide rich-semantic representation for the input audio. Yet, our framework produces efficient results with two-third number of epochs of what are required to train the model without our framework from scratch. We also conjecture that increasing the depth of feature extractor can mitigate this issue as shown in the case of WavLM model (see Table 2) where our framework maintains its superiority over the baseline $\text{RD}_{\text{w-sc}}$ [16].

3.2. Ablations

3.2.1. Framework performance evaluation as a function of training time

To establish the efficiency of proposed framework, we compute WER for different encoder sized models over the course of training, i.e. at epoch 25, 50, 75 and 100. Table 3 demonstrates that our framework surpasses the baseline models (referred as RD_{sc} and RD_{LD} in Table 1) at epoch 50 that is $\leq \frac{1}{3}$ of the training time required for the baseline methods. This also illustrates that embedding's alignment during training yields fast convergence and improves overall performance.

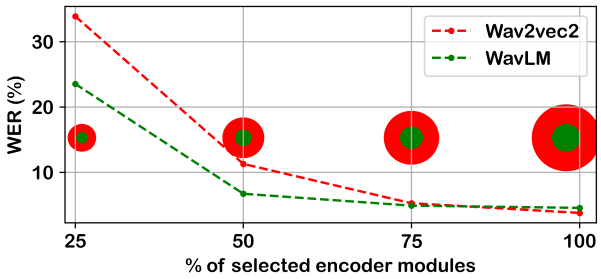

Fig. 2. Comparing dynamic versions WavLM (random dropping based) vs Wav2Vec2 (learnable masking based). Circle depicts the utilized parameters for different encoder sizes.

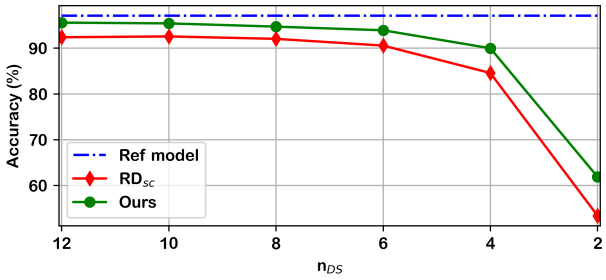

Fig. 3. Evaluating generalization capability of our framework on spoken language understanding task using FSC dataset. Baseline Accuracy (with full model $\mathcal{M}_{\text{ref}}$ = 97.1%)

3.2.2. Generalization of proposed framework on other downstream tasks

To signify the generalizability of our approach, we adapted the conformer architecture for spoken language understanding task, and evaluated it on fluent speech commands (FSC) [27] dataset. For the reference model, we trained the conformer architecture from scratch achieving a baseline accuracy of 97.1% which is in-line with original paper. Similarly, we trained the conformer using RD with $p_d = 0.5$ and our KD-based framework. Figure 3 presents the accuracy for different value of n_{DS}, signifying that effectiveness of DLD is not limited to ASR downstream task only, rather it can be applied to different applications yielding improved performance.

4. CONCLUSION

This work introduces an effective framework exploiting KD for curating dynamic speech architectures. Proposed DLD framework harnesses the rich semantic knowledge of teacher network embodied in its latent embeddings to supervise the learning of dynamic student network. The student network learns to dynamically adapt to different encoder sizes in parallel to minimizing the difference between embedding's distribution, hence producing effective latent embeddings. Extensive experimentation using conformer and wavlm architectures underlines the superior performance-computation trade-off as compared to state-of-the-art methods.

5. ACKNOWLEDGMENTS

This work was supported by Ministero delle Imprese e del Made in Italy (IPCEI Cloud DM 27 giugno 2022 - IPCEI-CL-0000007) and European Union (Next Generation EU).

6. REFERENCES

[1] Yann LeCun, John Denker, and Sara Solla, "Optimal brain damage," *Advances in neural information processing systems*, vol. 2, 1989.

[2] Chen Zhao, Yixiao Ge, Feng Zhu, Rui Zhao, Hongsheng Li, and Mathieu Salzmann, "Progressive correspondence pruning by consensus learning," in *ICCV*, 2021, pp. 6464–6473.

[3] Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *CVPR*, 2018, pp. 2704–2713.

[4] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[5] Karthik Kumar, Jibang Liu, Yung-Hsiang Lu, and Bharat Bhargava, "A survey of computation offloading for mobile systems," *Mobile networks and Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[6] Matsubara et al., "Split computing and early exiting for deep learning applications: Survey and research challenges," *ACM Computing Surveys*, pp. 1–30, 2022.

[7] T. Teerapittayanon, B. McDanel, , and H.T. Kung, "BranchyNet: Fast inference via early exiting from deep neural networks," *arXiv:1709.01686*, 2017.

[8] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger, "Deep networks with stochastic depth," in *ECCV*. Springer, 2016, pp. 646–661.

[9] Angela Fan, Edouard Grave, and Armand Joulin, "Reducing transformer depth on demand with structured dropout," *arXiv preprint arXiv:1909.11556*, 2019.

[10] Zuxuan Wu et al., "BlockDrop: Dynamic Inference Paths in Residual Networks," in *CVPR*, 2018.

[11] Minjia Zhang and Yuxiong He, "Accelerating training of transformer-based language models with progressive layer dropping," *NeurIPS*, pp. 14011–14023, 2020.

[12] Salah Zaiem, Robin Algayres, Titouan Parcollet, Slim Essid, and Mirco Ravanelli, "Fine-tuning strategies for faster inference using speech self-supervised models: a comparative study," in *ICASSPW*. IEEE, 2023, pp. 1–5.

[13] Abdul Hannan, Alessio Brutti, and Daniele Falavigna, "LDASR: An experimental study on layer drop using conformer-based architecture," in *EUSIPCO*, 2024.

[14] Zhourong Chen, Yang Li, Samy Bengio, and Si Si, "You look twice: Gaternet for dynamic filter selection in CNNs," in *CVPR*, 2019, pp. 9172–9180.

[15] Yifan Peng et al., "I3D: Transformer architectures with input-dependent dynamic depth for speech recognition," in *ICASSP*. IEEE, 2023, pp. 1–5.

[16] Abdul Hannan, Daniele Falavigna, and Alessio Brutti, "Input conditioned layer dropping in speech foundation models," *arXiv preprint arXiv:2507.07954*, 2025.

[17] David Genova, Philippe Esling, and Tom Hurlin, "Keep what you need: extracting efficient subnetworks from large audio representation models," in *ICASSP*, 2025.

[18] Abdul Hannan, Alessio Brutti, Shah Nawaz, and Mubashir Noman, "An Effective Training Framework for Light-Weight Automatic Speech Recognition Models," in *Interspeech 2025*, 2025, pp. 3613–3617.

[19] Jingjing Xu et al., "Dynamic encoder size based on data-driven layer-wise pruning for speech recognition," in *InterSpeech*, 2024, pp. 4563–4567.

[20] Ji Won Yoon et al., "Tutornet: Towards flexible knowledge distillation for end-to-end speech recognition," *IEEE/ACM TASLP*, pp. 1626–1638, 2021.

[21] Yevgen Chebotar and Austin Waters, "Distilling knowledge from ensembles of neural networks for speech recognition.," in *Interspeech*, 2016, pp. 3439–3443.

[22] Xiaoyu Yang, Qiujia Li, Chao Zhang, and Philip C Woodland, "Knowledge distillation from multiple foundation models for end-to-end speech recognition," *arXiv preprint arXiv:2303.10917*, 2023.

[23] Alex Graves et al., "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006, pp. 369–376.

[24] Sanyuan Chen et al., "WavLM: Large-scale self-supervised pre-training for full stack speech processing," *IEEE JSTSP*, pp. 1505–1518, 2022.

[25] Panayotov et al., "LibriSpeech: an ASR corpus based on public domain audio books," in *ICASSP*. IEEE, 2015.

[26] François Hernandez et al., "TED-LIUM 3: Twice as much data and corpus repartition for experiments on speaker adaptation," in *SPECOM*, 2018, pp. 198–208.

[27] Loren Lugosch, Mirco Ravanelli, Patrick Ignoto, Vikrant Singh Tomar, and Yoshua Bengio, "Speech model pre-training for end-to-end spoken language understanding," in *Interspeech*. 2019, pp. 814–818, ISCA.