Meta-Whisper: Speech-Based Meta-ICL for ASR on Low-Resource Languages

1st Ming-Hao Hsu
Electrical Engineering
National Taiwan University
Taipei, Taiwan
hsuminghao1006@gmail.com

2nd Kuan Po Huang
Computer Science and Information Engineering
National Taiwan University, AICS ASUS
Taipei, Taiwan
f09922005@ntu.edu.tw

3rd Hung-yi Lee
Electrical Engineering
National Taiwan University
Taipei, Taiwan
hungyilee@ntu.edu.tw

***Abstract*—This paper presents Meta-Whisper, a novel approach to improve automatic speech recognition (ASR) for low-resource languages using the Whisper model. By leveraging Meta In-Context Learning (Meta-ICL) and a k-Nearest Neighbors (KNN) algorithm for sample selection, Meta-Whisper enhances Whisper's ability to recognize speech in unfamiliar languages without extensive fine-tuning. Experiments on the ML-SUPERB dataset show that Meta-Whisper significantly reduces the Character Error Rate (CER) for low-resource languages compared to the original Whisper model. This method offers a promising solution for developing more adaptable multilingual ASR systems, particularly for languages with limited resources.**

***Index Terms*—In-Context Learning, Whisper, Multilingual ASR, Low-Resource Languages**

I. Introduction

Automatic speech recognition (ASR) systems have been developed for many years, models were primarily designed for single-language ASR tasks. These single-lingual ASR systems were tailored to recognize and transcribe speech in a specific language, often focusing on well-resourced languages like English, Mandarin, or Spanish. Early models, such as the Hidden Markov Model (HMM)-based systems [1]–[3], relied heavily on large datasets and hand-crafted features to achieve acceptable accuracy. Over time, as deep learning techniques like Recurrent Neural Networks (RNNs) [4]–[7] and Transformer-based architectures [8]–[12] emerged, the performance of single-lingual ASR systems improved significantly. These advancements allowed ASR models to achieve lower character error rates (CER) and word error rates (WER), particularly in common languages. However, these single-lingual systems were limited in their ability to generalize beyond the languages they were trained in, necessitating the development of separate models for each language. This inefficiency led to exploring multi-lingual ASR systems that could handle multiple languages within a single framework. Recent research in multilingual ASR has shown significant performance improvements, especially for low-resource languages. Multilingual models trained on multiple languages outperform monolingual baselines, with relative WER reductions of up to 28.8% [13]. These models can be language-agnostic, using a unified grapheme set [14] or accepting language identifiers as input [15]. Data augmentation techniques further enhance performance [16]. End-to-end architectures, such as sequence-to-sequence models, are well-suited for multilingual ASR [17]. Recent advancements include handling code-switching [18] and dialectal variations [19]. Novel approaches like semantic dataset creation using sentence embeddings show promise in improving multilingual ASR without language identification [20]. These developments contribute to more efficient and accurate multilingual ASR systems, which are particularly beneficial for low-resource languages and real-time applications.

Recent research on ASR for low-resource languages has explored various approaches to improve performance. Data augmentation techniques, including synthetic data generation and transliteration, have shown promising results [21]–[23]. Transfer learning from high-resource languages and unsupervised domain adaptation have been effective in enhancing ASR systems for low-resource targets [24], [25]. Multilingual training approaches have demonstrated improvements over monolingual systems [26]. Deep learning architectures, such as LSTM-CTC and Transformer models, have been successfully applied to low-resource ASR tasks [23], [26], [27]. Despite these advancements, ASR for low-resource languages remains challenging, with WERs ranging from 40% to 65% in severely constrained conditions [28]. Continued research and initiatives like the OpenASR Challenge [28] are crucial for advancing low-resource ASR technology.

To address this, we introduce **Meta-Whisper**, a novel framework aimed at enabling Whisper [29] to perform ASR tasks on low-resource languages using minimal paired speech and text examples. By employing a Meta In-Context Learning (Meta-ICL) [30] technique, we equip Whisper with the capability to handle ASR for these languages. Initially, we fine-tune Whisper on a set of common languages, teaching it how to perform ICL. Additionally, we enhance sample selection through a KNN sampling method, which extracts the hidden state representations from Whisper for more effective sampling. This approach differs from the procedure described in [31], which calculates Euclidean distances from Whisper's audio embeddings for ICL samples. In [31], they focus on Whisper's ICL abilities, demonstrating notable reductions in WER when adapting to Chinese dialects, all without the need for gradient-based training.

Our main contributions are twofold:

- We propose **Meta-Whisper**, a model capable of performing ASR on low-resource languages without direct training in those languages.
- Our approach, Meta-ICL, is highly efficient, requiring only 8 common languages with 10 minutes each to achieve significant performance improvements.

II. Methodology

A. Meta In-Context Learning (Meta-ICL)

In-context learning (ICL) refers to an emergent ability of large language models that the model can learn the analogy of the given examples and the target [32]–[39]. Then, the model can make inferences based on these examples. The speech-based ICL can be formulated as follows:

$$\hat{Y}_t = \arg\max P(Y_t|X, Y_{<t}, \mathcal{M}) \tag{1}$$

, where $\mathcal{M}$ is the given language model, X is the input speech, including example audio and target audio, and $Y_{<t}$ is the text sequence up to the current token, including starting tokens and example tokens.

Meta-ICL refers to how we make the model learn how to do ICL. Some models may not be familiar with the ICL data format. Therefore, some fine-tuning techniques may be necessary for the model to boost its ICL capability further. In our approach, we select some common languages in the Whisper model that already perform well. Using these languages to teach the Whisper model how to do ICL would be more effective. In our Meta-ICL training setting, we will give the Whisper model an example, including example speech, example text, and target speech. As Eq. (1) shows, the Whisper will make the next token prediction based on the given condition. However, we have a different loss calculation method from the regular fine-tuning of the Whisper model. We only calculate the cross entropy loss of the target tokens since we only need the Whisper model to learn how to make predictions based on the given condition instead of doing whole sequence prediction. This can be formulated as follows:

$$\mathcal{L}(\theta) = -\sum_{t=1}^{T} \log P(Y_t|X, Y_{<t}, \mathcal{M}_\theta) \tag{2}$$

, where $\mathcal{L}(\theta)$ is the loss function dependent on the model's parameters θ, T is the length of the target sequence, $P(Y_t|X, Y_{<t}, \mathcal{M}_\theta)$ is the probability of the next token Y_t given the input speech X, the previous tokens $Y_{<t}$, and the model's parameters $\mathcal{M}_\theta$. We randomly select the example for Meta-ICL training for each step to boost the generalization ability further. Therefore, the model knows how to make predictions based on different conditions for the same target. This can be represented as a sampling process $(X_i, Y_i) \sim \mathcal{D}$, where $\mathcal{D}$ is the training dataset, and (X_i, Y_i) is the randomly selected example for training step i, and the language of (X_i, Y_i) would be the same as the target audio.

We employ the Adaptive Low-Rank Adaptation (AdaLoRA) [40] technique for parameter-efficient fine-tuning of the models. This method allows us to fine-tune large models with fewer trainable parameters by focusing on low-rank updates to the weight matrices. For training, we selected the languages: English (eng), French (fra), German (deu), Russian (rus), Swahili (swa), Swedish (swe), Japanese (jpn), and Mandarin Chinese (cmn), as these are widely spoken and Whisper performs well on ASR tasks for these languages. In our training pipeline, we randomly select sample data from the training set to represent ICL. This random sampling during training enhances the model's generalization ability. For each selected sample-target pair, we extract the audio features separately using the Whisper encoder, concatenate them, and pad the features to a uniform length of 30 seconds. The concatenated audio features and the corresponding tokens of the sample audio are then fed into the model. During the iterative process, the model generates output tokens for each timestamp until it reaches the `<|endofsequence|>` token. The loss is calculated solely based on the model's predicted tokens without considering the original sample tokens. This encourages the model to learn from the context rather than rely on the audio data itself.

B. k-Nearest Neighbor In-Context Learning

When performing ICL, we aim to enhance the model's performance by providing Whisper with a similar sample to facilitate the learning process. We first input the audio A_i to the fine-tuned Whisper model $\mathcal{M}$ and extract its final layer hidden states as representations. This can be expressed as: $h_i = \mathcal{M}(A_i)$, where $\mathcal{M}(A_i)$ represents the representation from the Whisper model for audio A_i. Once the hidden state representations h_i are computed for each input audio A_i, we cache these representations for later comparison: $\mathcal{H} = \{h_1, h_2, \ldots, h_n\}$, where $\mathcal{H}$ is the set of cached hidden state representations for all n candidate audios.

Then, we choose the most similar audio for each target audio A_{target} based on calculating the KL divergence of the target audio representation h_{target} and each cached candidate audio representation h_i.

$$D_{\text{KL}}(h_{\text{target}} \parallel h_i) = \sum_j h_{\text{target},j} \log\left(\frac{h_{\text{target},j}}{h_{i,j}}\right) \tag{3}$$

The candidate audio with the smallest KL divergence to the target audio is considered the most similar and thus the best sample for in-context learning:

$$\hat{A} = \arg\min_{A_i} D_{\text{KL}}(h_{\text{target}} \parallel h_i) \tag{4}$$

, where $\hat{A}$ is the selected candidate audio that is most similar to the target audio.

III. Experimental Setup

A. Dataset

We utilized the ML-SUPERB dataset for both training and evaluation purposes. This dataset supports 143 languages,

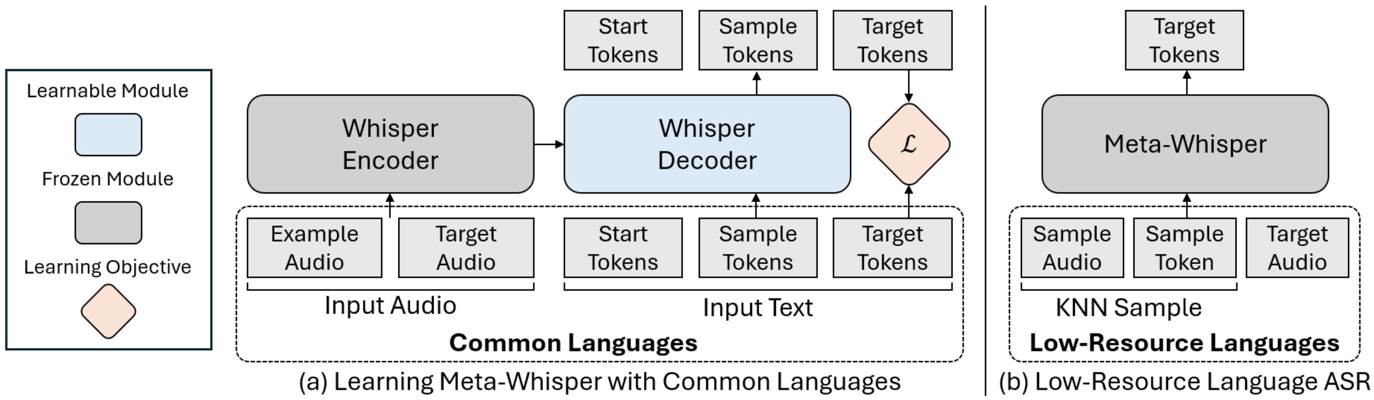

Fig. 1. **Meta-Whisper.** Our framework enables the Whisper to perform ASR on unseen low-resource languages. (a) **Learning Meta-ICL Whisper**: we use Whisper model objective $\mathcal{L}$ for Meta-ICL Whisper fine-tuning. $\mathcal{L}$ computed based on the predicted target tokens and ground truth target tokens. (b) **Low-Resource Language ASR**: we perform ASR tasks on low-resource languages with a KNN sampling mechanism. The KNN mechanism is based on KL divergence of audios' representations.

spanning from high-resource to endangered languages, making it an ideal choice for multilingual ASR tasks. The dataset is divided into several splits for each language: `train_10min`, `train_1hr`, `dev_10min`, and `eval_10min`. For our experiments, we selected the `train_10min` split for Meta-ICL training and the `eval_10min` split for evaluation. In our study, we categorize **low-resource languages** as those not pre-trained by Whisper. Based on this definition, 89 of the ML-SUPERB languages have been pre-trained by Whisper, leaving 54 languages classified as low-resource in this dataset.

B. Whisper Model

We selected the Whisper-large-v2 model as the backbone for our experiments. This is a large-scale multilingual ASR model pre-trained on 99 languages and consisting of 1.55 billion parameters. The Whisper model employs an encoder-decoder Transformer architecture, where the encoder processes the log-mel spectrogram of the input speech. At the same time, the decoder begins with start tokens and incorporates speech information into its Transformer layers using cross-attention mechanisms. The decoder then autoregressively predicts the next token in the sequence. We can perform speech-based ICL with the pre-trained Whisper model by leveraging this architecture and following a specific format.

C. Evaluation

We define the **sample audio** as the audio example used as context in the ICL process, while the **target audio** refers to the audio for which we aim to perform ASR. To compare these two audio samples, we calculate their similarity based on audio representations. We compute audio representations for each input using the Whisper encoder, leveraging a KNN approach. For sample selection, we use the `train_10min` split instead of the `test_10min` split, as we assume that transcriptions for the test data are unavailable. The higher the similarity between the sample audio and the target audio, the better the selected sample will improve ASR performance on the target. Initially, we use the Whisper encoder to extract input features from the sample and target audio. These features are then passed through the Whisper decoder without transcription, and the final audio representation is taken from Whisper's last hidden layer.

We calculate the KL divergence between each target audio and each audio in the training set to identify the most similar sample. Additionally, we modify the predicted output from the Whisper model to compute the CER. We also exclude outliers from the final calculation to ensure accurate results. These outliers typically consist of repeated tokens up to the maximum token limit, which can distort the average CER and fail to reflect the model's actual performance. As a result, we eliminate these outliers from our evaluation.

IV. Results

We define two evaluation methods: **Regular Evaluation** and **ICL Evaluation**. In **Regular Evaluation**, the target audio is used as the audio input, and start tokens are provided as the text input, allowing Whisper to predict the next token. In **ICL Evaluation**, we concatenate the sample audio with the target audio, using this combined audio as Whisper's input. The text input is formed by concatenating the start tokens with the sample tokens, enabling Whisper to predict the next token for the target audio, conditioned on the sample audio. Additionally, we define three language subsets: **low-resource languages (LRL)**, **Whisper pre-trained languages (WPL)**, and **Meta-ICL training languages (MTL)**. **LRL** refers to the low-resource languages not used during Whisper's pre-training. **WPL** denotes the languages that were part of Whisper's multilingual pre-training. **MTL** consists of the languages used for fine-tuning in the Meta-ICL process. In our experiment, we focus on the results of LRL subset.

A. Performance Across Different Approaches

In Table I, we present the CER for various approaches. The **Vanilla** model refers to the original Whisper model without any fine-tuning. The table highlights that our method achieves the lowest CER, outperforming the original Whisper

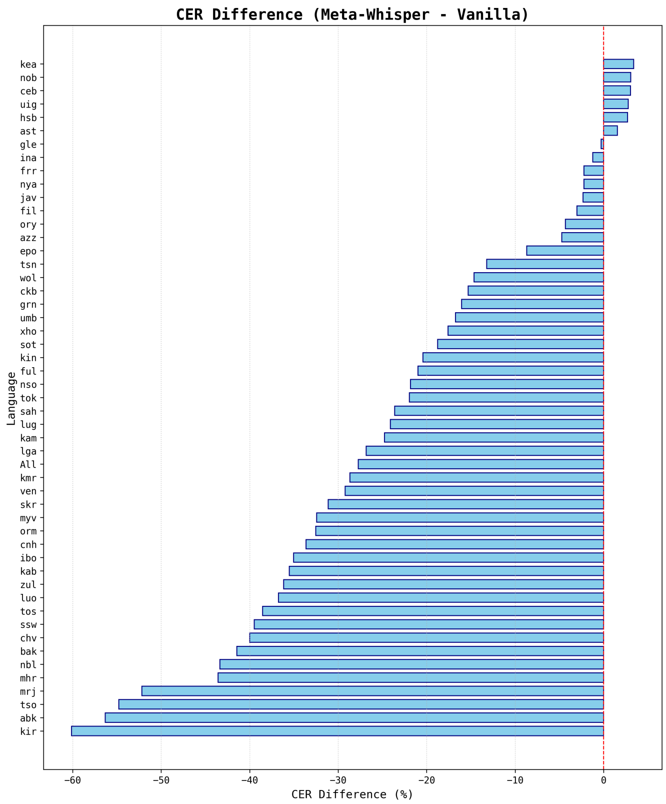

Fig. 2. CER comparison for Low-Resource Languages (LRL) between the Meta-Whisper and Vanilla Whisper models. Meta-Whisper outperforms Vanilla Whisper in 45 out of 54 low-resource languages.

model. Specifically, our approach attains a CER of 46.57%, significantly lower than the 74.29% observed with the Vanilla model, both with and without ICL, which yielded CERs of 57.72% and 74.29%, respectively. For Whisper's pre-trained languages (WPL), our method performs slightly less effectively, as fine-tuning on a particular language type impacts the overall performance of the Whisper model. However, the CER of the original Whisper model on the WPL subset is 31.76%, demonstrating that the KNN mechanism contributes positively to this task. Additionally, our method achieves the lowest CER among all models for the subset of languages used in Meta-ICL training, illustrating that it enhances model performance in these training languages. As shown in Fig. 2, Meta-Whisper outperforms the Vanilla Whisper model without ICL in 45 out of 54 low-resource languages.

TABLE I
CER comparison across various approaches.

Model	LRL	WPL	MTL
Vanilla w/o ICL	74.29%	59.47%	8.42%
Vanilla w/ ICL	57.72%	**31.76%**	6.98%
Meta-Whisper	**46.57%**	64.15%	**4.34%**

TABLE II
CER of Meta-Whisper fine-tuned on a different sampling mechanism. We report the performance of Meta-Whisper with different sampling methods.

Model	LRL
Meta-Whisper w/ Random Sample	51.33%
Meta-Whisper w/ KNN Sample	46.57%

TABLE III
The table shows the CER of our approaches trained on different dataset splits.

Model	LRL
Meta-Whisper w/ 1 sample	46.57%
Meta-Whisper w/ 2 samples	51.47%
Meta-Whisper w/ 3 samples	79.05%

B. Number of Samples Experiments

In the experiment, we aimed to assess the impact of the number of samples on Whisper's performance. In our main experiments, we used only one sample for ICL, as we observed that providing too many samples degraded Whisper's performance. We believe this decline may stem from difficulties in the increasing complexity due to longer input audio and text sequences. As illustrated in Table III, fewer samples lead to better performance. Specifically, when only one sample is provided for ICL, the CER is 46.57%. However, when two or three samples are used, the CER increases to 51.47% and 79.05%, respectively.

C. k-Nearest Neighbor Sampling

This section shows the performance of using different ICL sample-selecting techniques. In our proposed method, we first calculate the representation from the last hidden state of the Whisper model and compare the target audio's representation. We use KL divergence to check how two audios are similar to each other. As the table shows, the performance of the model using the KNN technique has a CER of 46.57%, which is better than 51.33% of the random sample since if the representation is more similar, maybe the two audio will be transliterated to similar language tokens.

V. Conclusion

Our proposed Meta-Whisper demonstrates significant improvements in ASR performance for low-resource languages. By achieving a 46.57% CER compared to Vanilla Whisper's 74.29%, our approach offers a scalable and computationally efficient solution for handling low-resource languages. The success of our method, which requires only a few examples of common languages and focuses on teaching the model ICL, highlights its potential for developing more adaptable multilingual ASR systems. Our findings on the effectiveness of single, well-chosen samples and the KNN-based selection method further underscore the efficiency of our approach. Meta-Whisper thus presents a promising path forward in expanding ASR capabilities to a wider range of languages, particularly those with limited resources.

REFERENCES

[1] S. Sakti and S. Nakamura, "Statistical speech recognition," 2009.
[2] N. Mathai, "A Literature Survey of Speech Recognition and Hidden Markov Models," 2009.
[3] J. Bilmes, "What HMMS CAN'T DO," 2004.
[4] A. L. M. et al., "Recurrent Neural Networks for Noise Reduction in Robust ASR," *Interspeech*, 2012.
[5] C. Weng and D. Yu, "Recurrent deep neural networks for robust speech recognition," *ICASSP*, 2014.
[6] W. Chan and I. Lane, "Deep Recurrent Neural Networks for Acoustic Modelling," *ArXiv*, 2015.
[7] O. V. et al., "Revisiting Recurrent Neural Networks for robust ASR," *ICASSP*, 2012.
[8] N. Moritz and T. Hori, "Streaming Automatic Speech Recognition with the Transformer Model," *ICASSP*, 2020.
[9] M. A. Haidar and C. Xing, "Transformer-based ASR Incorporating Time-reduction Layer and Fine-tuning with Self-Knowledge Distillation," *Interspeech*, 2021.
[10] T. H. et al., "Transformer-Based Long-Context End-to-End Speech Recognition," *Interspeech*, 2020.
[11] C. G. et al., "Pre-Training Transformer Decoder for End-to-End ASR Model with Unpaired Text Data," *ICASSP*, 2021.
[12] E. T. et al., "Transformer ASR with Contextual Block Processing," *ASRU*, 2019.
[13] V. P. et al., "Massively multilingual asr: 50 languages, 1 model, 1 billion parameters," *Interspeech*, 2020.
[14] A. D. et al., "Language-agnostic multilingual modeling," *ICASSP*, 2020.
[15] S. T. et al., "Multilingual speech recognition with a single end-to-end model," *ICASSP*, 2017.
[16] C. L. et al., "Multilingual asr with massive data augmentation," *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages and Collaboration and Computing for Under-Resourced Languages*, 2020.
[17] J. G.-D. et al., "A real-time end-to-end multilingual speech recognition architecture," *IEEE Journal of Selected Topics in Signal Processing*, 2015.
[18] S. A. C. et al., "Towards one model to rule all: Multilingual strategy for dialectal code-switching arabic asr," *Interspeech*, 2021.
[19] A. D. et al., "Multilingual and code-switching asr challenges for low resource indian languages," in *Interspeech*, 2021.
[20] O. H. A. et al., "Crossing language identification: Multilingual asr framework based on semantic dataset creation & wav2vec 2.0," *Machine Learning with Applications*, 2023.
[21] A. R. et al., "Low resource multi-modal data augmentation for end-to-end asr," *ArXiv*, 2018.
[22] B. T. et al., "Synthetic data augmentation for improving low-resource asr," *WNYISPW*, 2019.
[23] S. K. et al., "Low resource ASR: the surprising effectiveness of high resource transliteration," in *Interspeech*, 2021.
[24] A. C. S. et al., "Unsupervised domain adaptation schemes for building asr in low-resource languages," *ASRU*, 2021.
[25] O. S. et al., "Building an asr system for a low-resource language through the adaptation of a high-resource language asr system: Preliminary results," in *ICNLSP*, 2017.
[26] J. Billa, "ISI ASR system for the low resource speech recognition challenge for indian languages," in *Interspeech*, 2018.
[27] H. B. S. et al., "Advances in low resource asr: A deep learning perspective," in *Workshop on Spoken Language Technologies for Under-resourced Languages*, 2018.
[28] K. P. et al., "Openasr20: An open challenge for automatic speech recognition of conversational telephone speech in low-resource languages," in *Interspeech*, 2021.
[29] A. R. et al., "Robust speech recognition via large-scale weak supervision," in *ICML*, 2023.
[30] S. M. et al., "Metaicl: Learning to learn in context," *NAACL*, 2021.
[31] S. W. et al., "Can whisper perform speech-based in-context learning?" in *ICASSP*, 2024.
[32] K.-W. C. et al., "Exploring in-context learning of textless speech language model for speech classification tasks," in *Interspeech*, 2024.
[33] O. R. et al., "Learning to retrieve prompts for in-context learning," *NAACL-HLT*, 2021.
[34] N. W. et al., "The learnability of in-context learning," *NeurIPS*, 2023.
[35] S. G. et al., "What can transformers learn in-context? a case study of simple function classes," *NeurIPS*, 2022.
[36] S. M. X. et al., "An explanation of in-context learning as implicit bayesian inference," *ICLR*, 2021.
[37] Q. D. et al., "A survey on in-context learning," *ArXiv*, 2022.
[38] Z. Lin and K. Lee, "Dual operating modes of in-context learning," *ICML*, 2024.
[39] J. R. et al., "Human curriculum effects emerge with in-context learning in neural networks," *ArXiv*, 2024.
[40] Q. Z. et al., "Adalora: Adaptive budget allocation for parameter-efficient fine-tuning." ICLR, 2023.