

# GENERATIVE SPEECH RECOGNITION ERROR CORRECTION WITH LARGE LANGUAGE MODELS AND TASK-ACTIVATING PROMPTING

Chao-Han Huck Yang, Yile Gu, Yi-Chieh Liu, Shalini Ghosh, Ivan Bulyko, Andreas Stolcke

Amazon, USA

## ABSTRACT

We explore the ability of large language models (LLMs) to act as speech recognition post-processors that perform rescoring and error correction. Our first focus is on instruction prompting to let LLMs perform these task without fine-tuning, for which we evaluate different prompting schemes, both zero- and few-shot in-context learning, and a novel “task activation” prompting method that combines causal instructions and demonstration to increase its context windows. Next, we show that rescoring only by in-context learning with frozen LLMs achieves results that are competitive with rescoring by domain-tuned LMs, using a pretrained first-pass recognition system and rescoring output on two out-of-domain tasks (ATIS and WSJ). By combining prompting techniques with fine-tuning we achieve error rates below the N-best oracle level, showcasing the generalization power of the LLMs.

**Index Terms:** large language model, N-best rescoring, instruction prompting, few-shot learning, in-context learning.

## 1. INTRODUCTION

Large-scale language models (LLMs) have exhibited outstanding performance on downstream tasks by conditioning on input information, including task descriptions (e.g., performing mathematical calculations) or a limited number of input-output pairs obtained from training text (e.g., goal-oriented demonstrations). This new capability of task-specific inference from contextual information has been referred to as “*in-context learning*” in Brown *et al.* [1]. More specifically, the ability to learn in-context has been reported in previous studies [2] of pretrained LLMs with over 100B parameters trained with an unsupervised auto-regressive objective. Although recent advances in in-context learning have consistently demonstrated excellent performance on a wide range of tasks [3], there have been limited studies on the interaction or benefits of in-context learning on automatic speech recognition (ASR) tasks. As an example, contextual information [4] has been shown to play a vital role on ASR applications in complex domains, such as recognizing utterances referring to trending news.

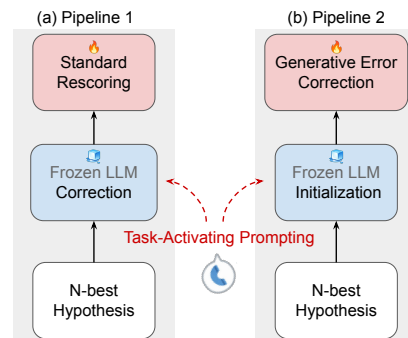
One open question in the development of robust ASR applications is *how* recent in-context learning frameworks can

utilize their zero-shot learning capability to enhance ASR systems. Meanwhile, scaling ASR model sizes up to 10B parameters [5] by itself has not proven adequate for achieving high performance on challenging (e.g., conversational) speech tasks from domain-specific data. The challenge to obtain better generalization of neural ASR models has motivated proposals to incorporate external knowledge from textual data [6]. For instance, one way to improve the RNN-transducer is to incorporate an external LM [7] for domain-aware adaptation in streaming-based applications. However, the external LM size is often limited to a range of 10M to 100M for on-device deployment. Given these limitations, cloud-based second-pass rescoring with LLMs may be a promising approach that leverages frozen pretrained models and leverages in-context learning.

Toward this end, in this work we explore novel ASR post-processing pipelines that utilize frozen LLMs by exploiting in-context learning. We consider two ASR second-pass pipelines, as shown in Figure 1:

**Pipeline 1:** a standard rescoring system takes in N-best output from a first ASR pass, and is trained to minimize the word error rate (MWER) by reranking the hypotheses. As illustrated in Figure 1(a), an LLM in-context learning process is inserted into the pipeline to post-process first-pass hypotheses to apply error correction.

**Pipeline 2:** a new task-activating prompting method is used to initialize the frozen LLM with task-oriented instruc-



**Fig. 1.** Two ASR post-processing frameworks using LLMs: (a) correct errors (e.g., grammar [8]) before applying a standard rescoring model, or (b) perform zero/few-shot rescoring; with optional task-activating prompting (Section 3.2).

tions. A list of N-best ASR hypotheses is formatted as input to the LLM, thus allowing “in-context learning initialization” and/or “in-domain fine-tuning” (e.g., using adapters for parameter-efficient model update) that results in an improved speech transcription.

In the remaining sections we present a first exploration of this novel way to utilize LLMs for the ASR task, demonstrate its surprising effectiveness, and compare results with different in-context learning schemes, as well as those of standard rescoring methods.

## 2. RELATED WORK

**LLM-based post-processing to improve hypotheses.** Error correction post-processing [9, 10] aims to fix grammar or deletion errors in output sentences and has been shown to improve the first-pass hypotheses generated from end-to-end ASR. A key characteristic of correction techniques is their reliance on pretrained LLMs, which benefit from rich contextual information. Liao *et al.* [9] propose ASR post-processing for readability, by extracting semantic expressions and generating readable text from ASR transcriptions. N-best T5 [11] used the T5 encoder-decoder architecture for rescoring with discriminative training.

**Zero-shot learning for acoustic and language modeling.** Prior work has demonstrated that language modeling can generalize to zero-shot multi-tasks without exemplars [3, 12, 13]. However, zero-shot and few-shot language modeling techniques often rely on the fine-tuning, which requires redeployment of pretrained models.

**In-context learning based on information prompting.** In-context learning (ICL) [1, 14] induces a single model to perform domain-agnostic inference without fine-tuning by providing a single or few prompts, thus addressing the aforementioned limitations. Prior study [2] has shown the ground truth demonstrations impose smaller effect than the author expected and significant zero-shot performance improvement under ICL framework. It implies external information gain can be extracted from frozen pretrained LLMs itself, if we select correct prompting strategy. However, ICL has its own shortcomings regarding tasks of reasoning. The chain-of-thought (CoT) prompting [15] decomposes the reasoning tasks by providing models with a sequence of questions or prompts that gradually guide the model to make predictions for a target task. While employment of CoT prompting is attributed to few-shot setup, LLMs has been proven to be the zero-shot reasoner given the single and specific prompt [16]. In this work, we applied the above ICL techniques to ASR rescoring for the first time and empirically evaluate the their performance individually against the baseline.

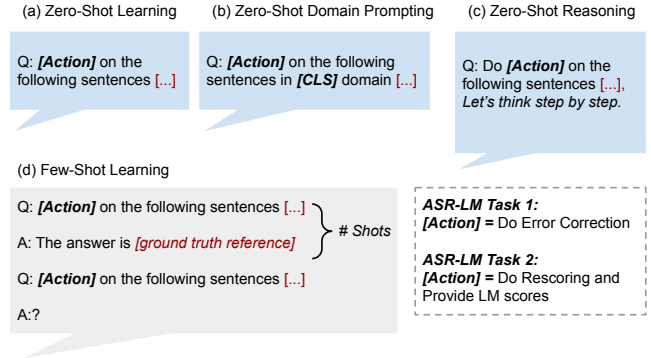


Fig. 2. Four LLM in-context learning uses for ASR 2nd pass

## 3. METHOD

We now review some recent advances in in-context learning techniques [15, 1, 16] and describe how they can be incorporated into second-pass rescoring applications.

### 3.1. In-Context Learning Background and Techniques

In-context learning [1] can emerge from modeling long-range coherence in the pretraining data. Based on a recent theoretical justification [17] by Bayesian inference, LLM would have implicitly learned to infer a latent concept during its pretraining stage. As an empirical result, in-context learning occurs if the LM can still infer the shared concept across examples (e.g., task instruction or prompts) to perform a target task. To model the in-context learning process, we can formulate its distribution over token  $o$  within the vocabulary  $O$  by sampling a latent *confounding variable* [18]  $\theta$  from its population  $\Theta$ .

The prediction over the pretraining distribution could be inferred by marginalizing over the confounding variable  $\theta$ :

$$p_{\text{prompt}} = p(o_1, \dots, o_T) = \int_{\theta \in \Theta} p(o_1, \dots, o_T | \theta) p(\theta) d\theta. \quad (1)$$

Under the in-context learning framework, prediction sequence  $O_i$  is inferred from the pretraining distribution conditioned on a prompt variable  $\theta^*$ , test-time sample (questions we would like to answer)  $x_{\text{test}}$  and its in-context predictor  $p_{\text{prompt}}(y|x)$ :

$$y_{\text{test}} \sim p_{\text{prompt}}(y|x_{\text{test}}, \theta^*). \quad (2)$$

For instance, a simple prompt to empower in-context learning is to directly provide a “*task-oriented question*” to the pretrained LLM, as shown in Figure 2(a). We further illustrate more in-context learning setups in the following subsections.

#### 3.1.1. Zero-shot domain-hint prompting

In the zero-shot setting, given a prompt template function  $r()$  and  $\theta^*$  as the domain-specific confounding variable (e.g., air-

line travel), a pretrained LLM models the conditional probability of the original input  $x$  and target  $y$ , even if they were never trained, into their template function  $r_x(x)$  and  $r_y(y)$ .

$$r_y(y_{\text{test}}) \sim p_{\text{prompt}}(r_y(y)|r_x(x_{\text{test}}), \theta^*). \quad (3)$$

In this work, we consider two general acoustic domains for making template function as a hard-coded input of “*airline information*” or “*financial market*”, as shown in Figure 2(b).

### 3.1.2. Zero-shot reasoning

Zero-shot reasoning [16] employs chain-of-thought prompting [15] in a zero-shot setting with only two prompts: (i) reasoning extraction and (ii) answer extraction. Based on [16], the reasoning extraction step uses a fixed and canonical prompt: *Let’s think step by step*, as shown in Figure 2(c).

In our experiments, we noticed that the reasoning extraction prompt is essential to boost the performance of zero-shot LLM rescoring. Under this zero-shot self-reasoning setup, the LLM output will first explain the task it is working on, then produce the actual task output. In the case of zero-shot rescoring, the LLM will first define a ASR-LM rescoring tasks and then provide LM scores for each N-best hypothesis.

### 3.1.3. Few-shot and one-shot in-context learning

A standard few-shot in-context learning process uses pairs of demonstrations [1] “questions and targeted tasks,” retrieved from training data, to inform frozen LLMs for performing the target output, as illustrated in Figure 2(d). One-shot in-context learning takes place by using a single demonstration as an input prompt for the frozen LLMs. Note that demonstrations (from an unseen training set) are distinct from test examples, and that the unsupervised-trained LLM has been reported to have a memory bottleneck based on term frequencies [19], which avoids potential data leakage issues for its few-shot learning evaluation reported in previous work [2, 1, 19].

### 3.1.4. N-best hypotheses to transcription fine-tuning

We introduce a hypotheses to transcription (H2T) mapping loss function:  $\mathcal{L}_{\text{H2T}} = \sum_{i=1}^N -\{\log P(y^*|x_i, \Theta) + \lambda \cdot \text{MSE}(s_i, P(y^*|x^{(i)}, \Theta))\}$ , where  $P(y^*|x_i, \Theta)$  represents the probability of the true transcription ( $y^*$ ) given the  $i$ -th hypothesis ( $x_i$ ) and the model parameters ( $\Theta$ ). To integrate acoustic information, a regularization term using mean squared error (MSE) is applied to penalize the model when there is a significant discrepancy between the predicted probabilities and the posterior probability scores with a  $\lambda$  coefficient of 0.01.

Furthermore, we also consider parameter-efficient fine-tuning methods as in [6], which only update a small subset  $\theta \subset \Theta$  of total trainable parameters to avoid potential overfitting that would hurt the generalization of the LLM [20, 21].

## 3.2. Task-activating Prompting (TAP) Framework

We now introduce a new in-context learning strategy that triggers the necessary sequential concepts for the ASR rescoring task, by utilizing multiple-round contextual sequences [22]. This technique is referred to as “task-activating prompting” (TAP). In this configuration, the LLM is given leading questions to clarify the task it needs to perform. Following this, the model is instructed to provide an example and, ultimately, it is presented with the top-N hypotheses from which to generate the actual output for the task. In our experiments, we noted that LLMs are capable of producing lists of the top N predictions, made up of utterances with analogous pronunciations. This demonstrates that LLMs assimilate acoustic (e.g., lattice-level) information during their pretraining phase. We illustrate the queries and responses used for task-activating prompting in Figure 3.

We observed that the responses from InstructGPT vary slightly as a function of the random seed. In our experiments, we utilize the API key from OpenAI, and the prompts are fixed, except for the final query that contains the sample and test N-best lists. Recent work on ICL [23] focuses on the selection strategy for in-domain demonstration samples, which clearly could affect inference results. However, we leave demonstration selection for the N-best task to future work, and in our *few-shot* learning experiments manually select longer demonstration utterances, following [2].

## 4. EXPERIMENTS AND RESULTS

We conduct our experiments by evaluating the final word error rate (WER) of a second-pass rescoring ASR system. First, we employ pretrained ASR models to decode audio from a target domain and collect the top-10 hypotheses. Then, we compare different in-context learning-based rescoring pipelines, including  $\mathcal{P}_1$ , the LLM-based error correction to enhance a standard ASR rescoring process, and  $\mathcal{P}_2$ , a prompting scheme that enables the LLM to directly perform ASR rescoring, as shown in Figure 1.

### 4.1. Pretrained ASR and Rescoring Model Training

We select a conformer-based RNN-transducer [24, 25] (760M parameters) pretrained with 960 hours of Librispeech [26], 10k hours of the GigaSpeech [27], 24k hours of English data in VoxPopuli [28], and wav2vec2-based Libri-Light [29, 30] pretraining as the first-pass model to generate the hypothesis list. We follow the conformer architecture presented in [25]. The pretrained model achieves a WER of 2.45% (test-clean) and 5.45% (test-other) on LibriSpeech without external LM.

For  $\mathcal{P}_1$ , to obtain a standard neural LM for rescoring, we employ an ASR-LM training procedure similar to Rescore-BERT [31], which involves a two-stage process. The first stage is adaptation using a first-stage masked language model (MLM) [32] loss, and the second stage is to use minimum

**Q:** Do you know speech recognition?

**R:** Yes, I am familiar with speech recognition. Speech recognition, also known as automatic speech recognition (ASR) or speech-to-text, is the process of converting spoken language into text. This technology involves using algorithms and machine learning models to analyze and transcribe the acoustic features of spoken words and phrases. Speech recognition has many applications, including voice-controlled assistants, automated phone systems, and transcription services.

**Q:** Do you know language model for speech recognition?

**R:** Yes, I can provide you with some information about language models used in speech recognition. Language models are an essential component of Automatic Speech Recognition (ASR) systems. They help to predict the likelihood of a sequence of words, given the context. This helps the ASR system to better understand and transcribe spoken language into text. This separate model is typically more complex and powerful than the initial language model used for the transcription, and it is used to re-score the transcription based on the probability of the words occurring in the given context. The rescoring process involves taking the output of the initial language model, which is usually based on statistical methods such as Hidden Markov Models, and then applying a more advanced language model, such as a neural network-based language model, to generate a more accurate transcription. This is accomplished by re-ranking the possible transcriptions based on the probabilities assigned by the more advanced language model. Language model rescoring has been shown to significantly improve the accuracy of speech recognition systems, particularly in noisy or challenging environments where the initial language model may not perform well.

**Q:** Could you give a possible example of language model rescoring with some hypotheses?

**R:** Sure, here is an example of language model rescoring for ASR with 5-best hypotheses:

1. recognize speech with artificial intelligence.
2. recognized speech with artificial intelligence.
3. recognize speech with artificial intelligent.
4. reckon eyes speech with artificial intelligence.
5. recognize peach with artificial intelligence.

After rescoring, the model assigns a higher probability to the first hypothesis, which is the correct transcription of the input. The other hypotheses receive lower probabilities, reflecting their decreased likelihood given the improved language model. In this case, the top-ranked hypothesis after rescoring would be chosen as the final output: recognize speech with artificial intelligence

**Q:** Nice job, I will provide some examples as a demonstration from [target domain]. The 10-best hypothesis is:[hypotheses list from training set], and I would expect your output is: [corresponding transcription]. Following this example, could you report the true transcription from the following 10-best hypotheses?: [hypotheses list for inference]

**Fig. 3.** Queries (Q) and responses (R) for N-best evaluation and correction by task-activating prompting (TAP) of LLMs

word error rate (MWER) training [33, 34, 35]. We choose the Albert [36] *base-v2* model with 4.8 million trainable parameters, and apply rescoring [31] using the error-corrected hypotheses obtained with a frozen LLM.

For  $\mathcal{P}_2$ , we examine how, by suitable prompt construction, the frozen pretrained LLMs can be instructed to perform ASR rescoring. The specific LLM configurations used are described next.

## 4.2. Pretrained LLM Configurations

**GPT-2** (1.5B): From the Generative Pretrained Transformer (GPT) family of causal models we selected this one [1, 37] as our basic LLM for in-context prompting setup. This version of GPT-2<sup>1</sup> is 100x smaller than very large teacher models, such as BLOOM [38], making it much more suitable for real-world deployment. GPT-2 is trained primarily using Wikipedia [39] and Common Crawl [40].

**OpenLLaMA** (13B): an open collection of transformer-based, decoder-only, causal language models ranging from 1B to 13B parameters [41]. It is trained exclusively on the RedPajama [42] dataset; we have confirmed that our Linguistic Data Consortium eval sets are not included.

**BLOOM** (176B): the first open-source LLM trained on the public supercomputer provided by the French government [38], BLOOM is available for reproducible studies of LLMs

over 100B. The model is pretrained using a large public collection of 498 HuggingFace datasets [43], comprising 1.61 TB of text spanning 46 natural languages and 13 programming languages (our evaluation datasets are not included.)

**InstructGPT** (175B): an LLM created by training a GPT-3 [1] model through reinforcement learning from human feedback (RLHF) [44]. InstructGPT demonstrates improved zero-shot learning performance, benefiting from human knowledge transferred through RLHF, a process similar to student-teacher learning. InstructGPT is trained using human feedback without using open-domain data for evaluation.<sup>2</sup>

## 4.3. Target-Domain Datasets

We use the pretrained ASR model introduced in Section 4.1 to decode two public datasets, both in  $\mathcal{P}_1$  (post-processing error correction) and  $\mathcal{P}_2$  (pretrained LLM-based rescoring).

**Airline Travel Information System (ATIS)** [45] contains 4978 training and 893 utterances. ATIS comprises spoken queries for air travel information, such as flight times and availability.

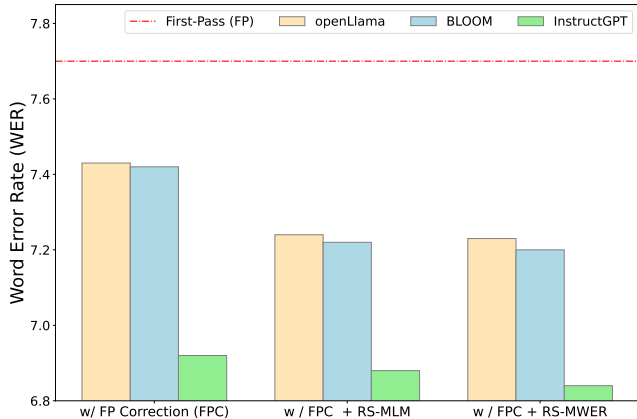
**Wall Street Journal (WSJ)** [46] consists of transcribed audio recordings of read news articles from the Wall Street Journal, covering a wide range of topics and featuring a diverse set of speakers. We adapt the pretrained conformer

<sup>1</sup>The pretrained model is publicly accessible under MIT License <https://github.com/openai/gpt-2>

<sup>2</sup>InstructGPT is an earlier version of ChatGPT. We did **not** use ChatGPT in our experiments due to its frequent revisions and unclear technical documentation.

**Table 1.** Oracle WERs for original and error-corrected N-best output, using  $\mathcal{P}_1$  processing as shown in Figure 1(a). The oracle error rates show the improvement in hypothesis quality as a result of post-processing using different sizes of LLMs.

| $\mathcal{P}_1$ : correction setup | WSJ         | ATIS        | WER <sub>avg</sub> | M-Size |
|------------------------------------|-------------|-------------|--------------------|--------|
| (a) $N$ -best                      | 9.78        | 6.43        | 8.11               | -      |
| (a) + corrected by GPT-2           | 9.91        | 6.11        | 8.01               | 1.5B   |
| (a) + corrected by OpenLLaMA       | 9.95        | 5.73        | 7.43               | 13B    |
| (a) + corrected by BLOOM           | 9.21        | 5.64        | 7.42               | 176B   |
| (a) + corrected by InstructGPT     | <b>8.41</b> | <b>5.43</b> | <b>6.92</b>        | 175B   |



**Fig. 4.**  $\mathcal{P}_1$  ASR rescoring (RS) training using hypotheses corrected by LLM. The dashed red line marks the  $N$ -best WER. The WER gradually decreases in the three stages of rescoring using our  $\mathcal{P}_1$  processing: Stage 0,  $N$ -best hypothesis with LLM correction ( $NC$ ); Stage 1, fine-tuned RescoreBERT [31] using the masked language modeling (MLM) loss; and Stage 2, MWER training.

model on the development set of *train-si284* and test on *93dev*.

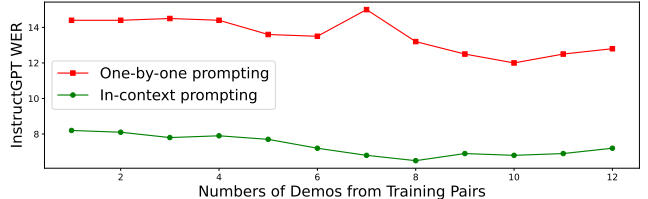
#### 4.4. Pipeline 1 Results

As shown in Table 1, we first use a pretrained LLM for error correction, using the setup in [9] to improve hypothesis quality as measured by oracle (minimum achievable)  $N$ -best error rates.

Figure 4 shows how error correction by LLMs complements existing rescoring with adapted LMs, such as RescoreBERT [31]. We observe that the existing rescoring pipeline [31] reduces WER from 11.3% to 8.7% compared to its fine-tuning-only baseline. Furthermore, Pipeline 1 employed the frozen pretrained language models to achieve an additional performance boost.

#### 4.5. Pipeline 2 Results

**Case 1: Zero-shot learning.** Table 3 shows results for rescoring with in-context learning using different LLMs, as well as various baselines. Note that for in-context rescoring, we ex-



**Fig. 5.** WER results on ATIS and WSJ with few-shot learning based on InstructGPT, for increasing numbers of demonstration samples. “One-by-one prompting” resets the model history after each utterance, “in-context prompting” lets the history (and thus the examples provided) accumulate.

tract LM scores from the model output responses, which differs from the standard approach of using the softmax outputs from the rescoring LM, for which we also report results. The best  $\mathcal{P}_2$  rescoring setup is the one using InstructGPT (Table 3, last row), achieving 19.7% relative WER reduction compared to rescoring with a fine-tuned GPT-2. Note that the frozen GPT-2 failed to give improvements over a 4-gram baseline, showing that a certain model size is required for generative error correction. For LLMs with over 100 billion parameters, the use of prompting information showed better results compared to using the softmax scores directly.

Next, we tested some popular prompting variants for in-context learning to possibly improve the performance of  $\mathcal{P}_2$ . As shown in Table 2,  $\mathcal{P}_2$  and prompting with “*step-by-step*” (known as zero-shot reasoning in [16]) achieved the best results for both LLMs (fourth row). It outperforms the one-shot learning variant (prompting with one input-output example; fifth row) by 1.8% relative WER difference. It is worth noting that the standard rescoring training  $\mathcal{P}_1$  (first row) still outperforms zero-shot  $\mathcal{P}_2$  by 14.6% relative.

**Case 2: Few-shot learning.** We gain some insight into the effects of in-context learning by considering few-shot learning in the form of conversational prompting. We feed InstructGPT examples drawn from the training portions of the two datasets, and report the results on the unseen test sets. As shown in Figure 5, we see that frozen InstructGPT improved its rescoring performance as the number of training samples is increased from 1 to 12. It is better to let the model history accumulate (green plot) than to reset it after each utterance (red plot), thereby compounding the effect of demonstrations.

**Case 3: In-domain fine-tuning.** We also obtained in-domain fine-tuning results, where we use the training portions of all the speech dataset to fine-tune the LLMs and then evaluate performance on the test sets; note that InstructGPT, being an API-only model, could not be fine-tuned. For prompting we use TAP (Section 3.2); however, we observed that after fine-tuning the exact method of prompting makes very little difference. As shown in Table 4, fine-tuning with low-rank adapters (LoRA) outperforms full fine-tuning in the generative error correction case, as do residual adapters. One reason would be that adapters avoid modifying the parameters

**Table 2.** WERs on ATIS and WSJ using prompting variants to enhance the  $\mathcal{P}_2$  in-context learning pipeline. We report the results of InstructGPT and BLOOM as LLMs over 100B; GPT-2 and OpenLLaMA do not perform consistently in this setting.

| In-context learning variant                                   | WSJ         |             | ATIS        |             |
|---|-------------|-------------|-------------|-------------|
|   | InstructGPT | BLOOM       | InstructGPT | BLOOM       |
| $\mathcal{P}_1$ : LLM-corrected $N$ -best w/ RescoreBERT [31] | 10.13       | 10.46       | 7.13        | 8.46        |
| $\mathcal{P}_2$ : (c) Zero-shot scoring                       | 10.43       | 11.23       | 7.95        | 8.45        |
| $\mathcal{P}_2$ : (c) + zero-shot reasoning [16]              | 10.20       | 11.88       | 7.77        | 8.53        |
| $\mathcal{P}_2$ : (c) + domain-hint prompting [2]             | 10.98       | 11.45       | 7.59        | 8.49        |
| $\mathcal{P}_2$ : (d) Scoring with one example-pair           | 9.42        | 9.45        | 6.34        | 7.30        |
| $\mathcal{P}_2$ : (d) + zero-shot reasoning [16]              | 9.87        | 11.46       | 7.25        | 8.64        |
| $\mathcal{P}_2$ : (d) + domain-hint prompting [2]             | 9.70        | 10.99       | 6.19        | 7.12        |
| $\mathcal{P}_2$ : (d) + task-activating prompting (TAP)       | <b>8.84</b> | <b>8.99</b> | <b>5.79</b> | <b>6.72</b> |

**Table 3.** WERs with  $\mathcal{P}_2$  pipeline, using LLM in-context learning (ICL) for rescoreing by zero-shot prompts illustrated in Figure 2(a). Where indicated we use task-activating prompting (TAP) to condition the LLM.

| $\mathcal{P}_2$ : zero-shot rescoreing setup | WSJ          | ATIS        |
|--|--------------|-------------|
| (a) Oracle                                   | 9.78         | 6.43        |
| (b) First pass                               | 11.87        | 8.82        |
| (b) + 4-gram LM                              | 11.21        | 8.57        |
| (b) + frozen GPT-2                           | 29.56        | 27.51       |
| (b) + frozen GPT-2 w/ TAP                    | 27.37        | 27.59       |
| (b) + frozen OpenLLaMA                       | 13.32        | 9.27        |
| (b) + frozen OpenLLaMA w/ TAP                | 11.53        | 8.61        |
| (b) + frozen BLOOM                           | 12.59        | 9.21        |
| (b) + frozen BLOOM w/ TAP                    | <b>10.82</b> | <b>8.42</b> |
| (b) + frozen InstructGPT                     | 9.97         | 7.15        |
| (b) + frozen InstructGPT w/ TAP              | <b>8.72</b>  | <b>6.39</b> |

of a pretrained model (by inserting a neural module with a small number of additional trainable parameters that approximate the full parameter updates), allowing for efficient learning of the task without affecting the pretrained parameters of the LLM. LoRA-based generative error correction introduces trainable low-rank decomposition matrices into the pretrained LLM layers, enabling the model to adapt to new data while keeping the original LLMs fixed to retain the pretrained knowledge. Specifically, LoRA performs a reparameterization of each model layer expressed as a matrix multiplication by inserting low-rank decomposition matrices. As a result, the representations (generated by the LLM) are not distorted due to task-specific tuning, while the adapter module acquires the ability to perform the error correction task.

Compared to previous state-of-the-art results with the Universal Speech Model (USM) [47] with text injection training over 2B parameters, our best fine-tuning results improve upon their WSJ results (WER of 3.2%). This is noteworthy considering that our generative error correction method is based on a smaller underlying conformer-RNN-T ASR system.

Another parameter-efficient form of fine-tuning is prefix tuning [50], where a continuous prompt prefix is inserted

**Table 4.** WERs on ATIS and WSJ, using fine-tuning (FT) and parameter-efficient adaptation to enhance the  $\mathcal{P}_{2-TAP}$  pipeline

| Method             | WSJ         |             |             | ATIS        |             |             |
|--------------------|-------------|-------------|-------------|-------------|-------------|-------------|
|                    | GPT-2       | Open-LLaMA  | BLOOM       | GPT-2       | Open-LLaMA  | BLOOM       |
| FT + ranking [11]  | 9.93        | 8.09        | 8.42        | 6.34        | 3.71        | 3.75        |
| Full fine-tune     | 9.94        | 7.71        | 6.91        | 5.98        | 2.23        | 2.49        |
| Res. adapter [48]  | <b>7.24</b> | 5.94        | 4.57        | <b>4.45</b> | 2.48        | 2.12        |
| LoRA [49]          | 7.52        | <b>2.11</b> | <b>2.81</b> | 4.57        | <b>1.69</b> | <b>1.97</b> |
| Prefix tuning [50] | 9.32        | 6.99        | 7.43        | 5.32        | 2.63        | 2.74        |

into the input and tuned to optimize task performance of the LLM. However, this method gave worse results than the full or adapter-based fine-tuning methods for the larger LLMs.

## 5. CONCLUSIONS

We have explored how in-context learning can be applied to pretrained large language models for improving first-pass ASR N-best output *without fine-tuning*. For this task setting, we introduce two post-processing pipelines utilizing in-context learning. The first one uses a pretrained LLM for error correction prior to standard rescoreing with a fine-tuned LM. The second pipeline uses in-context learning by prompting, to instruct the frozen pretrained LLM to perform the rescoreing task by itself. The latter method shows substantial gains over the first-pass ASR output and can be further enhanced with chain-of-thought and example prompting, as well as a new prompting scheme we call task-activating prompting. The best methods show 31% (on ATIS) to 38% (on WSJ) WER reduction over first-pass ASR, using a frozen InstructGPT, and better than with a fine-tuned GPT-2 LM. Substantial additional gains are achieved by fine-tuning the LLM for the ASR output-correction task. Post-processing with OpenLLaMA and LoRA fine-tuning achieves 86% and 80% WER reduction on ATIS and WSJ, respectively. These results are **below** the N-best oracle error rate, showing the LLM’s ability to utilize prelearned knowledge to correct ASR output errors. Possible future work can look at how to integrate extra acoustic representations into pretrained LLMs for further enhancing generative ASR error correction.

## 6. REFERENCES

- [1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 1877–1901, 2020.
- [2] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer, “Rethinking the role of demonstrations: What makes in-context learning work?” in *Proc. EMNLP*, 2022, pp. 11 048–11 064.
- [3] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, “Finetuned language models are zero-shot learners,” in *Proc. ICLR*, 2022.
- [4] R. D. Martinez, S. Novotney, I. Bulyko, A. Rastrow, A. Stolcke, and A. Gandhe, “Attention-based contextual language model adaptation for speech recognition,” in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 1994–2003.
- [5] B. Li, R. Pang, T. N. Sainath, A. Gulati, Y. Zhang, J. Qin, P. Haghani, W. R. Huang, M. Ma, and J. Bai, “Scaling end-to-end models for large-scale multilingual asr,” in *Proc. ASRU*. IEEE, 2021, pp. 1011–1018.
- [6] C. Chen, Y. Hu, C.-H. H. Yang, S. M. Siniscalchi, P.-Y. Chen, and E. S. Chng, “Hyporadise: An open baseline for generative speech recognition with large language models,” *arXiv preprint*, 2023.
- [7] D. Le, G. Keren, J. Chan, J. Mahadeokar, C. Fuegen, and M. L. Seltzer, “Deep shallow fusion for RNN-T personalization,” in *Proc. SLT*. IEEE, 2021, pp. 251–257.
- [8] J. Guo, T. N. Sainath, and R. J. Weiss, “A spelling correction model for end-to-end speech recognition,” in *Proc. ICASSP*. IEEE, 2019, pp. 5651–5655.
- [9] J. Liao, S. E. Eskimez, L. Lu, Y. Shi, M. Gong, L. Shou, H. Qu, and M. Zeng, “Improving readability for automatic speech recognition transcription,” *Transactions on Asian and Low-Resource Language Information Processing*, 2020.
- [10] J. Yang, R. Li, and W. Peng, “ASR error correction with constrained decoding on operation prediction,” *Proc. Interspeech*, pp. 3874–3878, 2022.
- [11] R. Ma, M. J. Gales, K. Knill, and M. Qian, “N-best T5: Robust ASR error correction using multiple input hypotheses and constrained decoding space,” *arXiv preprint arXiv:2303.00456*, 2023.
- [12] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja *et al.*, “Multitask prompted training enables zero-shot task generalization,” *arXiv preprint arXiv:2110.08207*, 2021.
- [13] R. Zhong, K. Lee, Z. Zhang, and D. Klein, “Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections,” *arXiv preprint arXiv:2104.04670*, 2021.
- [14] D. Dai, Y. Sun, L. Dong, Y. Hao, S. Ma, Z. Sui, and F. Wei, “Why can GPT learn in-context? language models implicitly perform gradient descent as meta-optimizers,” in *Proc. ICLR Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2023.
- [15] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. H. Chi, Q. V. Le, D. Zhou *et al.*, “Chain-of-thought prompting elicits reasoning in large language models,” in *NeurIPS*, 2022.
- [16] T. Kojima *et al.*, “Large language models are zero-shot reasoners,” *Advances in Neural Information Processing Systems*, 2022.
- [17] S. M. Xie, A. Raghunathan, P. Liang, and T. Ma, “An explanation of in-context learning as implicit bayesian inference,” *Proc. ICLR*, 2021.
- [18] C. Floccia, J. Goslin, F. Girard, and G. Konopczynski, “Does a regional accent perturb speech processing?” *Journal of Experimental Psychology: Human Perception and Performance*, vol. 32, no. 5, p. 1276, 2006.
- [19] Y. Razeghi, R. L. Logan IV, M. Gardner, and S. Singh, “Impact of pretraining term frequencies on few-shot reasoning,” *arXiv preprint arXiv:2202.07206*, 2022.
- [20] A. Kumar, A. Raghunathan, R. Jones, T. Ma, and P. Liang, “Fine-tuning can distort pretrained features and underperform out-of-distribution,” in *Proc. ICLR*, 2022.
- [21] C.-H. H. Yang, Y.-Y. Tsai, and P.-Y. Chen, “Voice2Series: Reprogramming acoustic models for time series classification,” in *Proc. International Conference on Machine Learning*. PMLR, 2021, pp. 11 808–11 819.
- [22] K. Hambardzumyan, H. Khachatrian, and J. May, “Warp: Word-level adversarial reprogramming,” in *Proc. ACL*, 2021, pp. 4921–4933.
- [23] X. Ye and G. Durrett, “Explanation selection using unlabeled data for in-context learning,” *arXiv preprint arXiv:2302.04813*, 2023.
- [24] C.-C. Chiu, T. N. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, E. Gonina *et al.*, “State-of-the-art speech recognition with sequence-to-sequence models,” in *Proc. ICASSP*. IEEE, 2018, pp. 4774–4778.
- [25] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, “Conformer: Convolution-augmented transformer for speech recognition,” *Proc. Interspeech*, pp. 5036–5040, 2020.
- [26] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “LibriSpeech: an ASR corpus based on public domain audio books,” in *Proc. ICASSP*. IEEE, 2015, pp. 5206–5210.
- [27] G. Chen, S. Chai, G. Wang, J. Du, W.-Q. Zhang, C. Weng, D. Su, D. Povey, J. Trmal, J. Zhang *et al.*, “GigaSpeech: An evolving, multi-domain ASR corpus with 10,000 hours of transcribed audio,” *arXiv preprint arXiv:2106.06909*, 2021.
- [28] 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,” *arXiv preprint arXiv:2101.00390*, 2021.
- [29] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen *et al.*, “Libri-Light: A benchmark for ASR with limited or no supervision,” in *Proc. ICASSP*. IEEE, 2020, pp. 7669–7673.

- [30] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 449–12 460, 2020.
- [31] L. Xu, Y. Gu, J. Kolehmainen, H. Khan, A. Gandhe, A. Rastrow, A. Stolcke, and I. Bulyko, “RescoreBERT: Discriminative speech recognition rescoring with BERT,” in *Proc. ICASSP*. IEEE, 2022, pp. 6117–6121.
- [32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *Proc. NAACL*, 2018, p. 4171–4186.
- [33] R. Prabhavalkar, T. N. Sainath, Y. Wu, P. Nguyen, Z. Chen, C.-C. Chiu, and A. Kannan, “Minimum word error rate training for attention-based sequence-to-sequence models,” in *Proc. ICASSP*. IEEE, 2018, pp. 4839–4843.
- [34] B.-H. Juang, W. Hou, and C.-H. Lee, “Minimum classification error rate methods for speech recognition,” *IEEE Transactions on Speech and Audio Processing*, vol. 5, no. 3, pp. 257–265, 1997.
- [35] D. Povey, “Discriminative training for large vocabulary speech recognition,” Ph.D. dissertation, University of Cambridge, 2005.
- [36] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, “ALBERT: A lite BERT for self-supervised learning of language representations,” in *Proc. ICLR*, 2020.
- [37] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, “Language models are unsupervised multitask learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
- [38] T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilić, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé *et al.*, “BLOOM: A 176b-parameter open-access multilingual language model,” *arXiv preprint arXiv:2211.05100*, 2022.
- [39] S. Merity, C. Xiong, J. Bradbury, and R. Socher, “Pointer sentinel mixture models,” in *Proc. ICLR*, 2018.
- [40] T. H. Trinh and Q. V. Le, “A simple method for commonsense reasoning,” *arXiv preprint arXiv:1806.02847*, 2018.
- [41] X. Geng and H. Liu, “OpenLLaMA: An open reproduction of LLaMA,” [https://github.com/openlm-research/open\\_llama](https://github.com/openlm-research/open_llama), May 2023.
- [42] T. Computer, “RedPajama: An open source recipe to reproduce LLaMA training dataset,” <https://github.com/togethercomputer/RedPajama-Data>, Apr. 2023.
- [43] Q. Lhoest, A. V. del Moral, Y. Jernite, A. Thakur, P. von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, L. Tunstall *et al.*, “Datasets: A community library for natural language processing,” in *Proc. EMNLP*, 2021, pp. 175–184.
- [44] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Gray *et al.*, “Training language models to follow instructions with human feedback,” in *Advances in Neural Information Processing Systems*, 2022.
- [45] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, “The ATIS spoken language systems pilot corpus,” in *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.
- [46] M. Marcus, B. Santorini, and M. A. Marcinkiewicz, “Building a large annotated corpus of English: The Penn Treebank,” *Computational Linguistics*, vol. 19, no. 2, 1993.
- [47] Y. Zhang, W. Han, J. Qin, Y. Wang, A. Bapna, Z. Chen, N. Chen, B. Li, V. Axelrod, G. Wang *et al.*, “Google USM: Scaling automatic speech recognition beyond 100 languages,” *arXiv preprint arXiv:2303.01037*, 2023.
- [48] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, “Parameter-efficient transfer learning for NLP,” in *International Conference on Machine Learning*. PMLR, 2019, pp. 2790–2799.
- [49] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in *Proc. ICLR*, 2021.
- [50] X. L. Li and P. Liang, “Prefix-tuning: Optimizing continuous prompts for generation,” in *Proc. ACL*, 2021, pp. 4582–4597.