Towards interfacing large language models with ASR systems using confidence measures and prompting

Maryam Naderi[1,2], *Enno Hermann*[1], *Alexandre Nanchen*[1], *Sevada Hovsepyan*[1], *Mathew Magimai.-Doss*[1]

[1]Idiap Research Institute, Martigny, Switzerland
[2]UniDistance Suisse, Brig, Switzerland

{maryam.naderi,enno.hermann,alexandre.nanchen,sevada.hovsepyan,mathew}@idiap.ch

Abstract

As large language models (LLMs) grow in parameter size and capabilities, such as interaction through prompting, they open up new ways of interfacing with automatic speech recognition (ASR) systems beyond rescoring n-best lists. This work investigates post-hoc correction of ASR transcripts with LLMs. To avoid introducing errors into likely accurate transcripts, we propose a range of confidence-based filtering methods. Our results indicate that this can improve the performance of less competitive ASR systems.

Index Terms: speech recognition, large language models

1. Introduction

Speech perception is a complex process that relies not only on acoustic information but also on environmental information, visual cues, context, and other factors. Consequently, speech perception in the brain is organized in a hierarchical and highly parallel processing network, where information on different time scales, about different linguistic units and from different modalities is analyzed to decipher the semantic content of speech [1]. Due to the reliance on these contextual cues during speech perception, humans can be considered "noisy listeners": to successfully understand the message, humans do not need to recognize every part of the speech they hear. Our predictive brain can replace the missing information based on the available contextual information [2, 3, 4, 5].

Automatic speech recognition (ASR) systems operate in a similar way. An acoustic model first processes the speech signal and identifies linguistic units, such as phonemes. Then, a language model (LM), which encodes prior knowledge about the likelihood of different word sequences, helps to find the most likely transcription given the potentially noisy information from the acoustic model.

The number of parameters in LMs and their performance on numerous benchmarks even without task-specific fine-tuning has increased so much in recent years that we commonly refer to them as large language models (LLMs). Especially instruction-tuned LLMs offer new possibilities for down-stream applications through their prompting mechanism [6]. LLMs can also work directly with very long input contexts, obviating the need to specifically adapt LMs to recently observed sequences [7].

Motivated by these developments, we investigate combining an ASR system with a LLM, where the latter is used as an additional LM to specifically address ASR errors. The primary challenge is the trade-off between leveraging LLMs to correct errors in low-accuracy transcripts while minimizing the risk of introducing new errors in more accurate ones. Figure 1 illustrates the proposed approach with an analogy to speech perception discussed before. Additionally, we evaluate the impact of LLM and ASR model size on the effectiveness of LLM corrections.

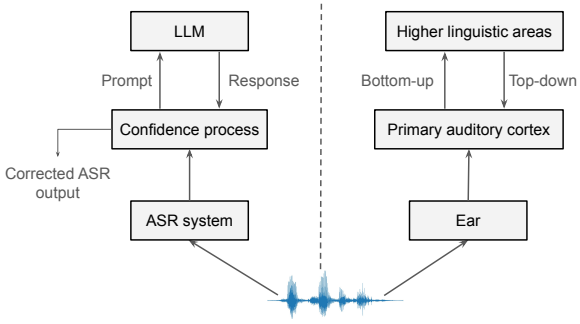

Figure 1: *Proposed approach (left) and speech processing in the brain (right).*

To reduce the chance of the LLM introducing new errors into the transcript, we propose three filtering methods that rely on the ASR confidence scores. For the first two, we let the LLM correct only sentences where the sentence or the lowest word confidence falls below a given threshold. For the third method, we prompt the LLM to only correct specific low-confidence words. To gain deeper insights into the LLM's behavior, we also present concrete examples where LLM has corrected errors in the transcription and other examples where LLM performed poorly.

The rest of this paper is organized as follows: In Section 2 we review previous works on applying LLMs to ASR. Section 3 details our experimental setup and in Section 4 we present our results and analysis.

2. Related works

With the advent of LLMs, a wide range of works have investigated how these could improve ASR performance. Generative instruction-tuned LLMs in particular offer new possibilities of combining ASR systems and LMs via prompting.

In hybrid speech recognition [8], the decoder returns a list or lattice of hypotheses by combining probabilities from the acoustic model and a basic n-gram language model. These hypotheses can then be rescored with a more powerful neural LM [9] or LLM [10]. Recent neural end-to-end ASR approaches directly learn an LM, but can also be combined with a separately trained one by shallow fusion or other methods [11].

In addition to traditional integration of LMs, one can focus specifically on identifying and correcting errors in ASR outputs [12]. Prior works framed post-hoc ASR error correction as a spelling correction or a machine translation problem [13, 14].

In traditional n-best rescoring, only the best hypothesis is selected, although another one could also be partially or fully correct. Chen et al. [15] therefore instruct LLMs to generate a new hypothesis based on all n-best options. They found that zero-shot prompting did not yield improvements on two datasets but adapting pre-trained LLMs with few-shot prompting, i.e. providing some example ASR outputs and corresponding corrections, and fine-tuning on a larger set of examples did.

Min et al. [16] explored the integration of LLMs in ASR systems to improve transcription accuracy. Their results show that directly applying the in-context learning capabilities of the LLMs for improving ASR transcriptions presents a significant challenge, and often leads to a higher word error rate (WER). However, other works [17, 18, 19] that explored the ability of LLMs to select, rescore and correct n-best list or ASR transcripts showed that zero and few-shot in-context learning can yield performance gains that are comparable to rescoring by domain-tuned LMs and can even achieve error rates below the n-best oracle level. Other works [20, 21] have also applied LLMs to spoken language understanding tasks, where the focus lies on identifying the correct intent from ASR transcripts, rather than correcting errors.

Our work, on the contrary, focuses on giving more insights on how to effectively use LLMs to improve ASR performance. In a few-shot, in-context learning scenario, we evaluate the ability of LLMs to correct ASR transcripts. Similar to Pu *et al.* [22], we propose filtering ASR outputs based on confidence scores to prevent the LLM from introducing errors into transcripts that are likely already correct. We further analyze the errors introduced and the corrections made by LLMs. By doing so, our work seeks to shed light on the strengths and limitations of LLMs, when applied to ASR.

3. Experimental setup

3.1. ASR system

We obtain initial ASR transcriptions from Whisper [23], a competitive set of models trained on 680,000 hours of transcribed speech. We run Whisper via the `whisper-timestamped` Python package [24], which supports extracting sentence- and word-level confidence scores. We compare the following models: *Tiny* (39M parameters), *Medium* (769M), *Large V3* (1550M). While English-only variants exist for the smaller models, we always use the multilingual one.

3.2. Large language model

We use the following OpenAI ChatGPT models as LLMs for error correction in the ASR transcriptions: `gpt-3.5-turbo-1106`, `gpt-3.5-turbo-0125`, and `gpt-4-0125-preview`. The GPT-3.5 turbo models have a context window of 16,385 tokens and can understand and generate natural language while GPT-4, with its 128,000 token context window, is a large multimodal model (handling both text and image inputs with text outputs) and tackles more complex problems more accurately than its predecessors [25]. In this work, we call ChatGPT via its Python API and do not modify any default parameters.

3.3. Confidence-based filtering

Passing every ASR output to the LLM for correction risks introducing errors into correctly transcribed sentences. To mitigate this, we compare different filtering methods based on confidence scores returned by the Whisper ASR model.

Whisper internally uses tokens that are obtained with byte-pair encoding [26]. Each output token is associated with a confidence score based on its log probability. The `whisper-timestamped` Python package [24] computes a word-level confidence by averaging all tokens that form a word and a sentence-level confidence by averaging all tokens in the sentence. Punctuation tokens are excluded in either case.

We then filter the ASR outputs that should be passed to the LLM based on the sentence-level or the lowest word-level confidence score in the sentence. We will refer to these two methods as *sentence-level* and *lowest-word* confidence. For sentences above a chosen confidence threshold, we retain the original ASR outputs.

As a third option, we prompt the LLM to only correct *specific words* in the ASR transcription that fall below a certain confidence threshold. If no words within the transcription fall below the confidence threshold, the original ASR transcription is retained.

3.4. Dataset

We evaluate our proposed approach on the English LibriSpeech corpus [27] of audiobook recordings. We use the `dev-clean` and `dev-other` subsets for initial experiments and hyperparameter tuning and then report final results on the `test-clean` and `test-other` evaluation sets. Each of these subsets contains around 2500–3000 utterances. Speakers in the `other` portions are more challenging to recognize and lead to higher WERs.

While these LibriSpeech subsets are not included in the Whisper training data, we cannot exclude that ChatGPT was trained on them due to the proprietary nature of the model.

4. Results

In this section, we present our results in terms of WER and character error rate (CER) on identifying a suitable prompt, comparing different ASR models, and filtering based on confidence scores. We also discuss and analyze the types of errors made by the LLM.

4.1. Prompt selection

We first describe our process of selecting a suitable prompt and analyze which elements of the prompt are important for ASR performance. LLMs perform best when the prompt contains a clear description of the task. For this reason, we provided information about the task, the format of the input and the expected output, and provided two examples in the prompt.

In the prompt, we clearly explain the task of correcting ASR errors to the LLM. We further describe the format of the input and expected response and instruct the LLM not to provide any explanatory or additional text besides the corrected transcription. We then provide one or two example input-output pairs for a few-shot learning scenario [15].

We show the base prompt for our experiments with a basic description of the task in Table 1. In other prompts, we explicitly instruct the LLM to make grammar corrections and to make changes that closely match the input transcription acoustically or phonetically. Results for these different prompts in Table 2 show that in particular providing more than one example and instructing to make phonetically plausible corrections improve the ASR performance. For all following experiments, we therefore use prompt 4.

Table 1: *LLM prompts used in this work. For certain experiments, the red and/or blue parts are added. Italic text shows examples provided after the system prompt, with the intended response in bold.*

> You are a helpful assistant that corrects ASR errors.
> You will be presented with an ASR transcription in json format
> with key: text and your task is to correct any errors in it.
> If you come across errors in ASR transcription, make corrections that closely match the original transcription acoustically or phonetically
> If you encounter grammatical errors, provide a corrected version adhering to proper grammar.
> Provide the most probable corrected transcription in string format.
> Do not change the case, for example, lower case or upper case, in the transcription.
> Do not output any additional text that is not the corrected transcription.
> Do not write any explanatory text that is not the corrected transcription.
>
> *Why not allow your silver tuff to luxuriate in a natural manner?*
> ***why not allow your silver tufts to luxuriate in a natural manner?***
> *Meanwhile, how fair did it with the flowers?*
> ***Meanwhile, how fared did it with the flowers?***
> `ASR transcription`

Table 2: *WER (%, lower is better) on LibriSpeech* `dev-clean` *of the original Whisper* `tiny` *output and corrections with* `gpt-3.5-turbo-1106` *for different prompts.*

Prompt	**WER**
Original ASR output	8.51
1: Base prompt (with one example)	7.49
2: Base prompt (with two examples)	6.76
3: 2 + do correct grammar mistakes	6.90
4: 2 + ensure corrections are phonetically similar	**6.65**
5: 2 + 3 + 4	6.79

4.2. Influence of ASR performance

We study the influence of ASR performance on the LLM corrections by comparing Whisper models of different strength. As shown previously [10], less competitive ASR models — Whisper *Tiny* and *Medium* in our case — leave more room for improvement.

We summarize the results in terms of WER and CER for the original ASR and the LLM-corrected transcripts (relative change in parentheses) on both development sets in Table 3. While we observe improvements in WER with LLM correction in most cases, the relative improvements in `dev-other` are smaller compared to the ones in `dev-clean`. This suggests that correcting errors in more difficult speech data also presents a greater challenge for the LLM model.

Table 3: *WER and CER of the original ASR output and LLM corrections with* `gpt-3.5-turbo-1106` *for different Whisper models (relative change in parentheses).*

Whisper model	**WER**			**CER**		
	ASR	+LLM (rel. (%))		ASR	+LLM (rel. (%))	
dev-clean						
Tiny	8.51	6.65	(-21.9)	3.49	3.08	(-11.7)
Medium	4.12	3.50	(-15.0)	1.79	1.42	(-20.7)
Large V3	3.11	3.34	(+7.4)	1.16	1.21	(+4.3)
dev-other						
Tiny	17.03	14.87	(-12.7)	8.16	7.71	(-5.5)
Medium	6.54	6.19	(-5.4)	2.96	2.90	(-2.0)
Large V3	4.62	4.59	(-0.6)	1.83	1.89	(+3.3)

4.3. Confidence-based filtering

In Figure 2 (left) we show the WERs for various *sentence-level* confidence thresholds for all Whisper models on the LibriSpeech `dev-clean` subset. Transcriptions with a confidence score higher than the threshold are not passed to the LLM for correction. The Figure shows that the optimal value for the threshold is 0.95 for *Tiny* and *Medium* models while the *Large* model is not sensitive to the threshold.

Figure 2 (right) shows the effect of varying the *lowest-word* confidence thresholds. A value of 0.7 provides a good trade-off of stable ASR performance and reducing the number of utterances that needs to be corrected by the LLM. We observed similar patterns for both methods on `dev-other`.

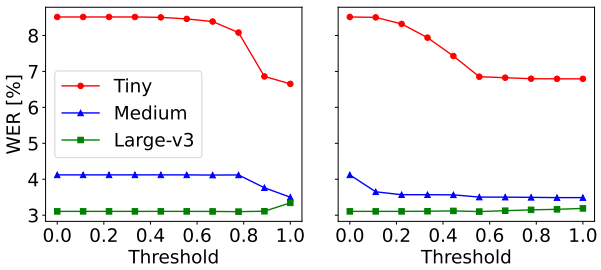

Figure 2: *WER for various sentence-level (left) and lowest-word (right) confidence thresholds for Tiny, Medium, and Large V3 Whisper models applied on* `dev-clean` *dataset with* `gpt-3.5-turbo-1106`.

4.3.1. Correction of specific words

In this experiment, we pass both the ASR transcriptions and a list of words with confidence scores below a predefined threshold to the LLM. [1] Figure 3 presents the WER results for various confidence thresholds.

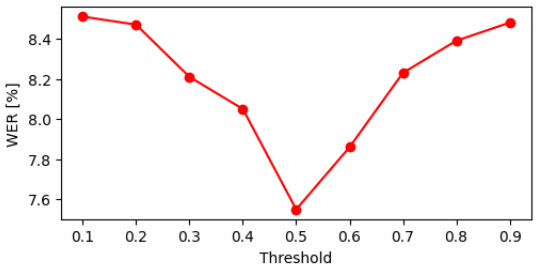

Figure 3: *WER for various thresholds for specific low-confidence words with Tiny Whisper model applied on* `dev-clean` *dataset with* `gpt-3.5-turbo-1106`.

As the figure demonstrates, thresholds close to 0 results in a WER near the original WER (without ChatGPT correction) but

[1] Replacing sentences 2–3 in prompt 4 from Table 2 with "You will be presented with an ASR transcription in json format with keys: `text` and `low_confidence_words`, where the `text` is the ASR transcription and `low_confidence_words` contains the list of words in the transcription with low confidence scores. Your task is to correct any errors in the transcription. If you come across errors in ASR transcription, make sure that you correct only words from within the `low_confidence_words` list and your corrections should closely match the original transcription acoustically or phonetically."

Table 4: *WER on the LibriSpeech test sets of the original ASR output and LLM corrections with* `gpt-3.5-turbo-0125`/`gpt-4-0125-preview` *for different Whisper models, comparing lowest-word and sentence-level confidence. For each case, we also show what percentage of utterances was passed to the LLM after confidence-thresholding.*

Whisper	**test-clean**				**test-other**			
model	ASR	GPT-3.5	GPT-4	% corrected	ASR	GPT-3.5	GPT-4	% corrected
Lowest-word confidence (threshold: 0.7)								
Tiny	8.13	6.55	5.65	86.6%	17.45	15.49	**13.65**	94.0%
Medium	4.27	**3.42**	3.54	64.3%	8.20	6.67	6.97	72.8%
Large V3	**2.78**	2.86	3.21	53.0%	**4.82**	4.91	4.93	60.4%
Sentence-level confidence (threshold: 0.95)								
Tiny	8.13	6.56	**5.63**	94.5%	17.45	15.51	13.67	98.0%
Medium	4.27	3.71	3.56	67.1%	8.20	6.77	**6.62**	79.8%
Large V3	**2.78**	2.83	3.13	48.4%	**4.82**	4.93	4.94	60.8%

a threshold of 1 gives a higher value for WER than the previous experiments where there was no low-confidence word list restriction. The WER reaches its lowest value of 7.55 at a threshold of 0.5, not matching the performance of the sentence-level and lowest-word confidence approaches.

4.3.2. Test set performance

Finally, we also present results for our selected best prompt and confidence thresholds on the LibriSpeech `test-clean` and `test-other` evaluation sets in Table 4. The best result for each dataset and Whisper model is highlighted in bold. Our findings indicate that, despite its higher number of parameters, GPT-4 only outperforms GPT-3.5[2] for Whisper *Tiny*, but does not result in additional improvements in WER for the transcriptions of the *Medium* and *Large* models.

4.4. Error analysis

In this section, we showcase examples of Whisper *Tiny* outputs on the development sets in which the LLM has corrected errors in the transcriptions, has failed to correct or even introduced new errors into the transcription.

(1) *REF: their fingers *** sear me like fire*
ASR: their fingers see her me like fire
LLM: their fingers *** sear me like fire

(2) *REF: damn your impertinence sir burst out burgess*
ASR: dam your impertinent sur burst out burges
LLM: damn your impertinent sir burst out burgess

(3) *REF: *** *** fedosya 's face made her anxious*
ASR: the dose used to face nature *** anxious
LLM: the dose used to face nature *** anxiously

Example 1 and 2 show cases where the LLM has corrected all or most of the errors within the ASR transcriptions of Whisper *Tiny*. Here, `REF`, `ASR`, and `LLM` denotes reference, ASR, and LLM-corrected transcriptions respectively. Example 3 is a typical case of where the LLM struggles to correct the transcript because it already contains too many errors and, for example, reconstructing proper nouns without acoustic context is challenging. Furthermore, Table 5 breaks down for how many utterances the LLM improved, worsened, or did not change the ASR performance. We find more improvements, but also more new errors on the more challenging `dev-other` subset.

Table 5: *Percentage of utterances where LLM improved, worsened, and did not change WER of Whisper Tiny outputs.*

Dataset	**Improved**	**Worsened**	**No Change**
`dev-clean`	26.38	4.85	68.78
`dev-other`	29.96	6.11	63.93

We also note that LLM corrections can sometimes decrease WER while increasing CER. This occurs because any number of character changes within a word only affects the WER by one unit ($\frac{1}{N}$ with N being the number of words in the reference transcription). However, the same changes can have a greater impact on CER.

(4) *REF: pour mayonnaise over all chill and serve*
ASR: parme a nays overall chill and serve
LLM: parmesan *** over all chill and serve

Example 4 demonstrates this effect. The LLM reduces WER from 57.14% to 28.57% in this example, while CER increases from 25.00% to 27.50%.

5. Conclusions

In this work we investigated LLMs for ASR error correction. Viewing ASR systems as noisy listeners, inspired by human speech perception, we proposed filtering ASR outputs based on confidence measures, so that the LLM only has to focus on less accurate transcripts. Indeed, our results confirm that LLMs especially boost ASR performance for less competitive acoustic models because otherwise there is little room left for improvement.

We plan to investigate additional confidence estimation methods and other ASR systems than Whisper in future work. LLM outputs could also be rescored again with the acoustic model to validate if the proposed changes are acoustically plausible. We will further consider other long-form datasets where utterances are not evaluated one-by-one and LLMs are expected to provide more benefits because of their long context windows. Finally, studies also need to be conducted on other languages, where LLMs might not perform as well as on English.

[2]Here we used `gpt-3.5-turbo-0125` which we found to perform similar to `gpt-3.5-turbo-1106` on the development sets, but is faster and more robust to API errors.

6. Acknowledgements

This research was partially funded by the Swiss Innovation Agency Innosuisse through the flagship project IICT (grant no. PFFS-21-47) and by the Swiss National Science Foundation through the Bridge Discovery project EMIL: Emotion in the loop - a step towards a comprehensive closed-loop deep brain stimulation in Parkinson's disease (grant no. 40B2–0_194794 EMIL). We thank Olivier Bornet for providing us with a ChatGPT API key and other technical support.

7. References

[1] J. P. Rauschecker and S. K. Scott, "Maps and streams in the auditory cortex: Nonhuman primates illuminate human speech processing," *Nature Neuroscience*, vol. 12, no. 6, pp. 718–724, Jun. 2009. [Online]. Available: http://www.nature.com/articles/nn.2331

[2] G. A. Miller and S. Isard, "Some perceptual consequences of linguistic rules," *Journal of Verbal Learning and Verbal Behavior*, vol. 2, no. 3, pp. 217–228, Oct. 1963. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022537163800870

[3] R. V. Shannon, F. G. Zeng, V. Kamath, J. Wygonski, and M. Ekelid, "Speech recognition with primarily temporal cues," *Science*, vol. 270, no. 5234, pp. 303–304, Oct. 1995. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/7569981

[4] L. Gwilliams, A. Marantz, D. Poeppel, and J.-R. King, "Top-down information shapes lexical processing when listening to continuous speech," *Language, Cognition and Neuroscience*, vol. 0, no. 0, pp. 1–14, 2023, publisher: Routledge _eprint: https://doi.org/10.1080/23273798.2023.2171072. [Online]. Available: https://doi.org/10.1080/23273798.2023.2171072

[5] E. Sohoglu, J. E. Peelle, R. P. Carlyon, and M. H. Davis, "Predictive Top-Down Integration of Prior Knowledge during Speech Perception," *Journal of Neuroscience*, vol. 32, no. 25, pp. 8443–8453, Jun. 2012. [Online]. Available: http://www.jneurosci.org/cgi/doi/10.1523/JNEUROSCI.5069-11.2012

[6] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe, "Training language models to follow instructions with human feedback," in *Proc. NeurIPS*, 2022.

[7] B. Krause, E. Kahembwe, I. Murray, and S. Renals, "Dynamic evaluation of neural sequence models," in *Proc. ICML*, 2018, pp. 2766–2775.

[8] H. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers, 1994.

[9] A. Deoras, T. Mikolov, and K. Church, "A Fast Re-scoring Strategy to Capture Long-Distance Dependencies," in *Proc. EMNLP*, 2011, pp. 1116–1127.

[10] T. Udagawa, M. Suzuki, G. Kurata, N. Itoh, and G. Saon, "Effect and analysis of large-scale language model rescoring on competitive ASR systems," in *Proc. Interspeech*, 2022, pp. 3919–3923.

[11] S. Toshniwal, A. Kannan, C.-C. Chiu, Y. Wu, T. N. Sainath, and K. Livescu, "A Comparison of Techniques for Language Model Integration in Encoder-Decoder Speech Recognition," in *Proc. SLT*, 2018, pp. 369–375.

[12] R. Errattahi, A. El Hannani, and H. Ouahmane, "Automatic Speech Recognition Errors Detection and Correction: A Review," *Procedia Computer Science*, vol. 128, pp. 32–37, 2018.

[13] J. Guo, T. N. Sainath, and R. J. Weiss, "A spelling correction model for end-to-end speech recognition," in *Proc. ICASSP*, 2019, pp. 5651–5655.

[14] O. Hrinchuk, M. Popova, and B. Ginsburg, "Correction of Automatic Speech Recognition with Transformer Sequence-To-Sequence Model," in *Proc. ICASSP*, 2020, pp. 7074–7078.

[15] 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," in *Proc. NeurIPS*, 2023.

[16] Z. Min and J. Wang, "Exploring the Integration of Large Language Models into Automatic Speech Recognition Systems: An Empirical Study," in *Proc. International Conference on Neural Information Processing (ICONIP)*, 2023, pp. 69–84.

[17] R. Ma, M. Qian, P. Manakul, M. J. F. Gales, and K. Knill, "Can Generative Large Language Models Perform ASR Error Correction?" *ArXiv*, vol. abs/2307.04172, 2023. [Online]. Available: https://arxiv.org/abs/2307.04172

[18] C.-H. H. Yang, Y. Gu, Y.-C. Liu, S. Ghosh, I. Bulyko, and A. Stolcke, "Generative Speech Recognition Error Correction With Large Language Models and Task-Activating Prompting," in *Proc. ASRU*, 2023, pp. 1–8.

[19] S. Radhakrishnan, C.-H. Yang, S. Khan, R. Kumar, N. Kiani, D. Gomez-Cabrero, and J. Tegnér, "Whispering LLaMA: A cross-modal generative error correction framework for speech recognition," in *Proc. EMNLP*, 2023, pp. 10 007–10 016.

[20] G. Li, L. Chen, and K. Yu, "How ChatGPT is Robust for Spoken Language Understanding?" in *Proc. Interspeech*, 2023, pp. 2163–2167.

[21] M. He and P. N. Garner, "Can ChatGPT Detect Intent? Evaluating Large Language Models for Spoken Language Understanding," in *Proc. Interspeech*, 2023, pp. 1109–1113.

[22] J. Pu and T.-S. Nguyen, "Multi-stage Large Language Model Correction for Speech Recognition," *ArXiv*, vol. abs/2310.11532, 2023, https://arxiv.org/abs/2310.11532.

[23] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proc. ICML*, 2023, pp. 28 492–28 518.

[24] J. Louradour, "whisper-timestamped," https://github.com/linto-ai/whisper-timestamped, 2023.

[25] J. Achiam *et al.*, "GPT-4 Technical Report," https://api.semanticscholar.org/CorpusID:257532815, 2023.

[26] R. Sennrich, B. Haddow, and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," in *Proc. ACL*. Association for Computational Linguistics, 2016, pp. 1715–1725.

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