🦜 Parrot: Autoregressive Spoken Dialogue Language Modeling with Decoder-only Transformers

Ziqiao Meng[1*†], Qichao Wang[2*], Wenqian Cui[1], Yifei Zhang[1], Bingzhe Wu[3], Irwing King[1], Liang Chen[2†], Peilin Zhao[3]
[1]The Chinese University of Hong Kong, [2]Sun Yat-sen University, [3]Tencent AI Lab

Abstract

Recent advancements in large language models (LLMs) have demonstrated significant potential in enhancing real-time spoken interactions. Presently, open-source methodologies predominantly depend on intermediate generative text-based transcriptions to manage real-time spoken dialogues. However, these techniques often struggle with providing seamless interactions that involve real-time streaming audio inputs. In this research, we unveil an innovative spoken dialogue language model, **Parrot**, distinguished by its unique pre-training and supervised fine-tuning (SFT) pipeline. This pipeline deviates from conventional methodologies by utilizing both single-channel audio data and dual-channel spoken dialogue data to train the textless speech language model. During pre-training, we transform single-channel audio input into a sequence of discrete tokens, thereby instructing the LLM to identify audio tokens via next-token predictions. In the SFT phase, we pioneer a novel approach to dual-channel generative spoken dialogue language modeling with a unique "next-token-pair prediction" objective, facilitating the LLM's comprehension of natural human conversations. Our pipeline equips LLM to produce spoken interactions that are more natural and fluid than those generated by baseline approaches, as substantiated by thorough evaluations.

1 Introduction

The advent of large language models (LLMs), particularly the GPT series [16, 13, 14], has profoundly transformed the field of artificial intelligence. Among these modalities, audio or speech data holds particular importance as it enables LLMs to engage in real-time voice interactions with humans. The recently unveiled GPT-4o model [14] exhibits a remarkable proficiency in managing real-time interactions with users in conversational contexts. Throughout the demo presentation, it was able to generate authentic emotional responses and engage users with swift reactions. These functionalities, however, introduce additional challenges, as the model must thoroughly interpret the distinct audio information within human speech while conducting inference with minimal delay.

In this study, we present a novel pre-training and supervised fine-tuning (SFT) pipeline to develop a robust model, referred to as **Parrot**, specifically designed for spoken dialogue language modeling. The pre-training phase begins with the conversion of continuous audio inputs into a sequence of tokens, a process made possible by training a vector-quantized autoencoder (VQVAE) [20] to reconstruct these audio signals. We then leverage pretrained LLMs as a foundation for continuous learning on *single-channel* audio sequences, with the goal of next-token prediction. This is accomplished by integrating the learned audio tokens into the original text vocabulary. This pretraining stage aids LLMs in capturing the primary latent distribution of audio token sequences. In the subsequent

[*]Equal contribution. [†]Corresponding authors.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

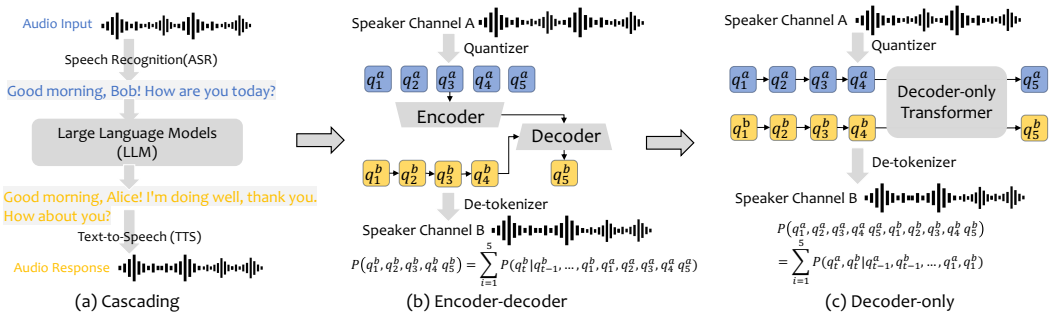


Figure 1: (a) The cascading approach depends on the intermediate text-based response generation translated by ASR and TTS; (b) The encoder-decoder spoken dialogue language modeling encode one of the speaker's audio sequence $Q^a = (q_1^a, q_2^a, ..., q_T^a)$ as condition information to decode another speaker sequence Q^b following the probability distribution $P(Q^b) = \sum_{i=1}^{T} P(q_t^b | q_{t-1}^b, ..., q_1^b, Q^a)$; (c) Our novel decoder-only spoken dialogue language modeling follows the newly proposed next-token-pair prediction paradigm such that $P(Q^a, Q^b) = \sum_{i=1}^{T} P(q_t^a, q_t^b | q_{t-1}^b, ..., q_1^b, q_{t-1}^a, ..., q_1^a)$.

stage, we utilize *dual-channel* audio data for SFT. The key advantage is enabling LLMs to directly comprehend how humans engage in natural dialogues. Unlike existing approaches, we introduce a novel "**next-token-pair prediction**" paradigm to model the dual-channel spoken dialogue generation using the decoder-only transformer. The comparison between our proposed method and existing techniques are illustrated in Figure 1. We carry out extensive experiments to validate the superiority of our innovative approach. Specifically, **Parrot** consistently outperforms strong baseline methods by 150% and 200% in average in terms of the reflective pause and interruption response accuracy respectively. Additionally, it achieves a low latency of 300ms.

2 Parrot: Training and Inference Pipeline

Our **Parrot** comprises two essential steps. The first involves pretraining the LLM on single-channel audio token sequences using the traditional "next-token prediction" objective. The second step fine-tunes the LLM on dual-channel audio token sequences, employing the innovative "next-token-pair prediction" paradigm. The rationale behind this strategy stems from the fundamental observation that the single-channel audio data can be sourced from the vast amount of open-source data available on the web. However, the primary limitation of single-audio data is its lack of speaker identity information and the overlapping regions between different speakers can be misleading. On the other hand, dual-channel spoken dialogue data encapsulates crucial turn-taking events with distinct speaker channels, and any overlapping event can be easily discerned.

2.1 Audio Tokenization and Single-channel Audio Pretraining

A single-channel audio is a continuous input sequence $\mathbf{x} \in \mathbb{R}^T$ with time length T. Owing to the high sampling rate of continuous audio signals, it is essential to employ an audio tokenizer, which extracts valuable features for the purpose of compressing the information. The audio quantizer $\mathcal{Q}$ projects the audio sequence $\mathbf{x}$ into a set of discrete tokens $Q = (q_1, ..., q_{T'}) = \mathcal{Q}(\mathbf{x})$ ($T' \ll T$), where each token q_t is an integer index from the vocabulary $q_t \in [V]$ where the vocabulary size is V. We train the audio tokenizer $\mathcal{Q}$ following the VQ-VAE [20] framework. In contrast to certain prior studies, we directly train the tokenizer on the raw audio signals $\mathbf{x}$, rather than transforming $\mathbf{x}$ into a mel-spectrogram first. We primarily adopt the training strategy presented in SoundStream [22], and provide a brief overview of its underlying mechanism.

Specifically, audio inputs $\mathbf{x}$ is fed into an encoder $\mathcal{E}$ to derive down-sampled latent features $\mathbf{f} \in \mathbb{R}^{\frac{T}{r} \times D}$ such that $\mathbf{f} = \mathcal{E}(\mathbf{x})$ with the down-sampling rate r and the latent dimension D. This is achieved by the CNN [9] architecture, which can capture the local dependency of $\mathbf{x}$. Then the quantizer $\mathcal{Q}$ converts the latent feature $\mathbf{f}$ to discrete tokens $\mathbf{q} \in \mathbb{R}^{\frac{T}{r}}$ such that $\mathbf{q} = \mathcal{Q}(\mathbf{f})$ where each entry q_i is a quantized integer index. Each latent feature $\mathbf{f}_i$ for time frame i is mapped to the code index q_i of its

nearest embedding vector in the Euclidean sense:

$$q_i =_{v\in[V]} \|\mathbf{z}_v - \mathbf{f}_i\|_2, \tag{1}$$

where $\mathbf{z}_i$ denotes the ith embedding vector of the learnable codebook $\mathbf{z} \in \mathbb{R}^{V\times D}$ containing $|V|$ vectors. Then the reconstructed audio signals $\hat{\mathbf{x}}$ are obtained through the decoder $\mathcal{G}$ such that $\hat{\mathbf{x}} = \mathcal{G}(\mathbf{z}_q)$ where $\mathbf{z}_q \in \mathbb{R}^{\frac{T}{r}\times D}$ denotes the codebook embedding vectors of the latent feature $\mathbf{f}$ indexed by $\mathbf{q}$. This autoencoder is trained by both the reconstruction loss and discriminator loss through straight-through estimators with stop-gradient operations. We direct readers to [22] for a comprehensive description of the architectures and algorithms involved.

After converting the input audio signals into the sequence of audio tokens Q, we subsequently supplement these audio tokens into the original LLM's text token vocabulary. Following this, we train the LLMs on the sequence Q using the standard autoregressive approach with the next-token prediction paradigm:

$$p(q_1, q_2, ..., q_{T'}) = \prod_{t=1}^{T'} p(q_t|q_{t-1}, .., q_2, q_1). \tag{2}$$

2.2 Supervised Fine-tuning with Dual-channel Audio

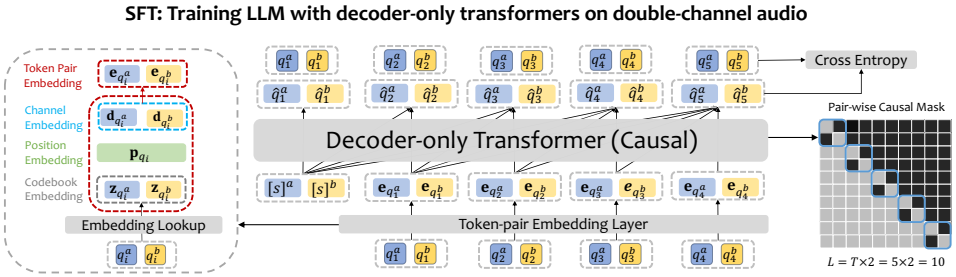


Figure 2: The illustration of the SFT learning mechanism of **Parrot** on the dual-channel spoken dialogue data. The novel architecture consists of two important modules. The first module is the embedding layer for obtaining the token-pair embedding; The second module is the decoder-only transformer with a *pair-wise causal masking* attention for next-token-pair prediction. $[s]^a$ and $[s]^b$ denote the special start tokens of channel a and channel b respectively.

The dual-channel audio input comprises a pair of time-aligned single-channel audio inputs, denoted as $(\mathbf{x}^a, \mathbf{x}^b)$, where each channel corresponds to a specific speaker. A fresh challenge arises in the generative modeling of dual-channel audio sequences using the decoder-only transformer architecture of LLMs. To address this issue, we propose a novel generative learning paradigm called *next-token-pair prediction*. The key idea here is to generate a sequence of time-aligned token pairs, rather than a single token, in an autoregressive fashion. In contrast to the conventional next-token prediction, our objective is more suitable to the generative modeling of an interpolated dialogue sequence which contain two separate channel identities. Specifically, we begin by discretizing both channels into time-aligned sequences with quantized audio tokens, denoted as $(Q^a = (q_1^a, q_2^a ..., q_T^a), Q^b = (q_1^b, q_2^b ..., q_T^b))$. To accommodate the input sequence structure within the decoder-only transformer architecture, we reorganize both sequences into a single interpolated dialogue sequence, represented as $Q^{\text{input}} = \{q_1^a, q_1^b, q_2^a, q_2^b, ..., q_T^a, q_T^b\}$. Subsequently, we model the probability distribution that generates the next token pair (q_t^a, q_t^b) at next time step t conditioned on the previously generated token pairs from step 1 to $t-1$:

$$p(q_1^a, q_1^b, q_2^a, q_2^b ..., q_T^a, q_T^b) = \prod_{t=1}^{T} p(q_t^a, q_t^b | q_{t-1}^a, q_{t-1}^b, ..., q_2^a, q_2^b, q_1^a, q_1^b). \tag{3}$$

Then we decompose the token pair conditional generating distribution $p(q_t^a, q_t^b | q_{t-1}^a, q_{t-1}^b, ..., q_1^a, q_1^b)$ by assuming the conditional independence between q_t^a and q_t^b:

$$p(q_t^a, q_t^b | q_{t-1}^a, q_{t-1}^b, ..., q_1^a, q_1^b) = p(q_t^a | q_{t-1}^a, q_{t-1}^b, ..., q_1^a, q_1^b) p(q_t^b | q_{t-1}^a, q_{t-1}^b, ..., q_1^a, q_1^b). \tag{4}$$

We illustrate this conditional independence and the dialogue distribution modeling in Figure 1. The probability distribution in Eq.3 and Eq.4 adheres to a fundamental inductive bias that *a person's speech is influenced by both his own previous statements and what he has heard in the past*. To adapt to the generative modeling of the newly arranged dialogue sequence Q^{input}, we need to modify the embedding layer and the attention masking mechanism accordingly. Our novel token-pair embedding layer consists of three important embeddings in total, which are codebook embedding $\mathbf{z}$, position embedding $\mathbf{p}$ and channel embedding $\mathbf{d}$. Specifically, for each token pair q_t^a, q_t^b:

$$\mathbf{z}_{q_t^a}, \mathbf{z}_{q_t^b} = \text{lookup}(\mathbf{z}, q_t^a, q_t^b), \quad \mathbf{p}_{q_t^a} = \mathbf{p}_{q_t^b}, \quad \mathbf{d}_{q_t^a}, \mathbf{d}_{q_t^b} = \text{one-hot-embedding}(\mathbf{id}^a, \mathbf{id}^b). \tag{5}$$

In the above Eq. 5, $\mathbf{d}_{q_t} \in \mathbb{R}^D$ denotes the channel embedding of its one-hot identity encoding $\mathbf{id}$, which indicates the speaker role (a or b) of token q_t. The positional encoding is represented as $\mathbf{p}_{q_t} \in \mathbb{R}^D$ indicating which time step both tokens are from. It is important to note that both q_t^a and q_t^b share the same positional embedding, with the Llama 3 [5] model utilizing the Rotary positional embedding as described in [18]. After the token-pair embedding layer, we obtain the input embedding $\mathbf{e}_{q_t} = [\mathbf{z}_{q_t}, \mathbf{p}_{q_t}, \mathbf{d}_{q_t}]$ for each token q_t (a or b). Following the implementation of Llama 3, we add both positional embedding and channel embedding to the query and key vectors (instead of value vectors) of each token pair as follows:

$$\mathbf{q} = \mathbf{W}_Q[\mathbf{z}_{q_t^a}, \mathbf{z}_{q_t^b}] + [\mathbf{p}_{q_t^a}, \mathbf{p}_{q_t^b}] + [\mathbf{d}_{q_t^a}, \mathbf{d}_{q_t^b}], \ \mathbf{k} = \mathbf{W}_K[\mathbf{z}_{q_t^a}, \mathbf{z}_{q_t^b}] + [\mathbf{p}_{q_t^a}, \mathbf{p}_{q_t^b}] + [\mathbf{d}_{q_t^a}, \mathbf{d}_{q_t^b}]. \tag{6}$$

Following the above Eq. 6, we obtain the query and key matrices for all token pairs, represented as $\mathbf{Q}, \mathbf{K} \in \mathbb{R}^{2T \times D}$, which are projected by weight matrices $\mathbf{W}_Q, \mathbf{W}_K$ respectively. Then we separately multiply codebook embedding vectors by $\mathbf{W}_V$ to obtain the value matrices $\mathbf{V} \in \mathbb{R}^{2T \times D}$. Based on these vectors, we conduct the attention computation as follows:

$$\mathbf{O} = \text{SoftMax}((\mathbf{Q}\mathbf{K}^T/\sqrt{D}) \cdot \mathbf{M})\mathbf{V}, \ \mathbf{M} \in \mathbb{R}^{2T \times 2T}. \tag{7}$$

The pair-wise causal masking matrix $\mathbf{M} \in \mathbb{R}^{2T \times 2T}$ is used to mask out the entries in the self-attention matrix, preventing each token q_t from attending to future tokens $(q_{t'}, t' > t)$ and simultaneously attending to tokens from another channel at the same time (i.e. q_t^a and q_t^b cannot attend to each other). The final layer output embedding, denoted as $\mathbf{O}^l \in \mathbf{R}^{2T \times 2T}$, is utilized to generate the next-token-pair prediction $(\hat{q}_{t+1}^a, \hat{q}_{t+1}^b)$ for each (q_t^a, q_t^b) via classifications over codebook embedding indices. The total training loss is equal to the sum of cross-entropy loss over all generated token pair predictions and the ground-truth token pairs. The overall modified embedding layers and self-attention layers are illustrated in Figure 2. Certain advanced architectural components present in Llama 3, such as grouped-queries attention and feedforward layers, have been omitted here, as our modifications do not impact them.

2.3 Streaming Inference

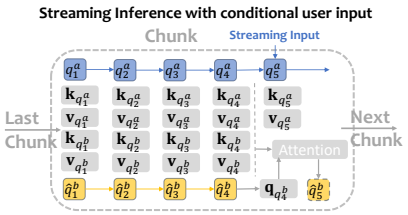


Figure 3: The figure illustrates the chunk-wise streaming inference process. Within each chunk, $(q_1^a, q_2^a, q_3^a, q_4^a, q_5^a)$ represents the provided speaker sequence.

In order to simulate a real-time user-assistant communication scenario, our speech LLM **Parrot** should be proficient in conducting conditional inference with streaming user voice input. In this inference setting, one speaker's voice input is provided as the user, and the model is assigned the task of inferring the other audio channel. This creates a situation that resembles a constrained generation problem. If the inference process strictly follows the training process, then the model should predict $\hat{q}_t^b$ immediately after receiving the speaker's voice input q_t^a at time t. However, due to the VQ-VAE audio tokenization mechanism, it's not feasible to receive just a single audio token from the speaker channel during the streaming inference. This is because the VQ-VAE requires a complete audio signal input within a specific time window. Therefore, unlike the training process, we need to determine when the model should start generating spoken responses upon receiving streaming user input audio tokens. Specifically, we adopt a divide-and-conquer approach to the inference process, breaking it down into chunks, each containing a predetermined number of tokens, denoted as λ. Each time the number of user input tokens reaches λ (a chunk of speaker input is given), our model begins to generate predictions until the number of predicted tokens also reaches λ (a chunk is filled). This procedure is repeated until the end of user voice inputs (e.g., the conclusion of the voice-assistant service). This inference process is illustrated in the accompanying Figure 3.

3 Experiment

3.1 Reflective Pause and Interruption evaluation

In this section, we leverage GPT-4 to meticulously craft 1k diverse conversational scenarios that reflect typical pauses and interruptions observed in natural dialogue. These scenarios are designed to capture the nuances and complexities of real-life interactions, providing rich evaluation settings for our analysis.

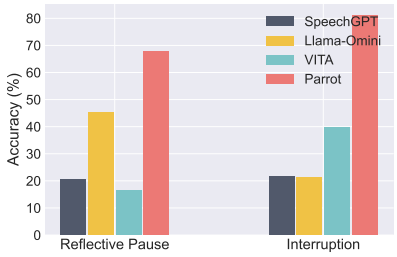


Figure 4: Interaction response accuracy.

For reflective pauses, **Parrot** demonstrated the highest accuracy at approximately 60%, significantly outperforming the other models. VITA and Llama-Omini followed with accuracies around 30% and 20%, respectively, while SpeechGPT lagged behind with an accuracy below 10%. This suggests that **Parrot** is particularly adept at managing the subtleties of reflective pauses in conversation, potentially due to its advanced contextual understanding capabilities.

Besides, **Parrot** excelled with an impressive accuracy of nearly 80%, indicating its robustness in handling abrupt conversational changes. VITA also performed relatively well, achieving an accuracy of around 55%. Both SpeechGPT and Llama-Omini showed lower accuracies, with SpeechGPT slightly outperforming Llama-Omini.

3.2 Quality and statistics of generated dialogues

Table 1: Linguistic quality and turn-taking statistics of generated dialogues, including the number of turn-Taking events and cumulative durations per minute, compared to the ground truth.

Model	**Number of occurrences / min**				**Cumulated duration /min**			
	ΔIPU	**ΔPause**	**ΔGap**	**ΔOverlap**	**ΔIPU**	**ΔPause**	**ΔGap**	**ΔOverlap**
dGSLM w/o CA	-3.9	0.9	-3.6	-1.	-12.1s	8.3s	-1.4s	2.5s
dGSLM	-1.6	3.4	-2.	-2.9	-4.6s	3.6s	0.8s	-1.9s
LSLM	-2.2	3.6	-2.4	-3.2	-4.1s	3.4s	-1.5s	-2.3s
Cascaded	-4.1	-7.	7.4	-6.5	1.3s	-5.5s	0.9s	-3.6s
Parrot$_{0.1}$	-1.4	2.1	-2.0	-1.	-3.2s	**2.5s**	-1.2s	-2.1s
Parrot$_{0.5}$	-1.5	**1.9**	-1.8	-1.5	**-2.9s**	3.0s	**-0.9s**	-2.2s
Parrot$_{0.9}$	**-1.3**	2.2	**-1.5**	**-0.9**	-3.3s	2.8s	-1.4s	**-1.9s**

We evaluate the linguistic quality and turn-taking dynamics of generated dialogues using various models, as detailed in Table 2. The detailed evaluation settings are in the A.7.2. LSLM[11] integrates speaker channels at the embedding layer and separates them in the final layer, demonstrates a notable reduction in the number of Inter-Pausal Units (IPUs) and gaps, indicating smoother transitions between speakers. The dGSLM[12], particularly with the cross-attention(CA) module, shows a significant decrease in the cumulative duration of pauses and gaps, suggesting more fluid and continuous dialogue. Comparatively, **Parrot** exhibit balanced performance with moderate reductions in both the number and duration of turn-taking events, highlighting their potential for generating natural and coherent dialogues. These findings underscore the importance of model architecture in optimizing dialogue flow and linguistic quality.

4 Conclusion

We propose a novel streaming spoken dialogue language model, **Parrot**, by employing a unique pretraining and supervised fine-tuning pipeline. We propose a novel algorithm to enable the decoder-only transformer to handle spoken dialogue in an autoregressive manner and we have successfully achieved the streaming audio response generation while the model is actively listening to users in real-time, outperforming the baseline cascading approaches. Based on experiments, our method significantly improves the naturalness and fluidity of generated spoken dialogues, addressing the limitations of existing text-based and encoder-decoder methods in real-time interactive scenarios.

5 Acknowledgements

Qichao Wang is supported by the National Key R&D Program of China under grant No. 2022YFF0902500, the Guangdong Basic and Applied Basic Research Foundation, China (No. 2023A1515011050), Shenzhen Science and Technology Program(KJZD20231023094501003)and Tencent AI Lab RBFR2024004.

References

[1] Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, Michael Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. *arXiv preprint arXiv:1912.06670*, 2019.

[2] Guoguo Chen, Shuzhou Chai, Guanbo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, et al. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio. *arXiv preprint arXiv:2106.06909*, 2021.

[3] Christopher Cieri, David Graff, Owen Kimball, Dave Miller, and Kevin Walker. Fisher english training speech part 1 transcripts. *Philadelphia: Linguistic Data Consortium*, 2004.

[4] Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue. Technical report, Kyutai, September 2024.

[5] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

[6] Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. Llama-omni: Seamless speech interaction with large language models. *arXiv preprint arXiv:2409.06666*, 2024.

[7] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[8] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P. E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux. Libri-light: A benchmark for asr with limited or no supervision. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7669–7673, 2020. `https://github.com/facebookresearch/libri-light`.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pages 1106–1114, 2012.

[10] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 11513–11522. IEEE, 2022.

[11] Ziyang Ma, Yakun Song, Chenpeng Du, Jian Cong, Zhuo Chen, Yuping Wang, Yuxuan Wang, and Xie Chen. Language model can listen while speaking. *CoRR*, abs/2408.02622, 2024.

[12] Tu Anh Nguyen, Eugene Kharitonov, Jade Copet, Yossi Adi, Wei-Ning Hsu, Ali Elkahky, Paden Tomasello, Robin Algayres, Benoît Sagot, Abdelrahman Mohamed, and Emmanuel Dupoux. Generative spoken dialogue language modeling. *Trans. Assoc. Comput. Linguistics*, 11:250–266, 2023.

[13] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[14] OpenAI. Gpt-4o system card, 2024.

[15] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5206–5210. IEEE, 2015.

[16] Ajay Patel, Bryan Li, Mohammad Sadegh Rasooli, Noah Constant, Colin Raffel, and Chris Callison-Burch. Bidirectional language models are also few-shot learners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[17] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[18] Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[19] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[20] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[21] Zhifei Xie and Changqiao Wu. Mini-omni: Language models can hear, talk while thinking in streaming, 2024.

[22] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE ACM Trans. Audio Speech Lang. Process.*, 30:495–507, 2022.

[23] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 15757–15773. Association for Computational Linguistics, 2023.

A Appendix

A.1 Detailed Related Work Discussions

We compare **Parrot** with several newly released speech LLMs, which are Mini-Omni [21], Llama-Omni [6], Moshi [4], LSLM [11].

1) Mini-Omni: The major advancement of this model is the batched parallel decoding strategy.
   - Advantages: Text generation can significantly enhance the quality of the audio produced. Concurrently, the implementation of batched parallel decoding can substantially mitigate issues related to inference latency. Overall, Mini-Omni effectively maintains a high standard of response quality while circumventing the latency typically associated with TTS translations.
   - Limitations: This model, while a multi-modal QA system, adheres to the standard architecture of multi-modal LLMs with various modality adaptors. However, it falls short in handling natural spoken conversations with real-time streaming user voice inputs. The dynamic nature of real-time dialogues, characterized by various pauses and turn-taking events, cannot be effectively simulated by this system.
2) Llama-Omni: This speech LLM also mainly focuses on enhancing the decoder stage like the previous Mini-Omni model. It propose an non-autoregressive decoder to simultaneously generate texts and audios. The text token is firstly upsampled and then fed into the speech decoder to derive the output voice. Unlike traditional TTS, Llama-Omni applies TTS word by word in an non-autoregressive manner.
   - Advantages: Like the Mini-Omni, this model also enjoys the response reliability due to the usage of intermediate text generation. In this way, Llama-Omni also enjoys low inference latency while maintaining high-quality content response.
   - Limitations: The Llama-Omni also shares the same limitations like Mini-Omin. Relying on text generations cannot handle special speech tokens that are hard to match to text tokens. In addition, the multi-modal LLMs can only handle multi-turn QA while failing to handle natural conversations like interruptions and pauses.
3) LSLM: This speech LLM explicitly leverages the double-channel audio data. Unlike **Parrot**, LSLM fuses two channel tokens into one single token and still follows the next-token prediction training objective. To enable LSLM to learn to interrupt, this work trains the speech LLM on the synthetic interruption data.
   - Advantages: No need to change the next-token prediction paradigm of the original LLM, which keeps the speech LLM as simple as possible.
   - Limitations: The introduction of the special “EOS" token and the “interruption" token will bring additional challenges in audio preprocessing. A threshold must be determined to filter what tokens are assigned to be “interruption token", which can be tricky. In addition, this model can only learn to interrupt by training on specific synthetic data. First, it might be troublesome to synthesize turn-taking events. Second, there is always a distribution gap between synthetic turn-taking and real-world turn-taking.
4) Moshi: This is a newly open-sourced speech LLM with high-quality spoken responses and minimal inference latency. Moshi leverages the RVQ technique to tokenize the audio inputs. And it explicitly proposes the usage of multi-channel audio modeling. There are mainly text channels, speaker audio channels and listener audio channels. The generative modeling of the multi-channel token sequences is following the RQtransformer [10], which is an encoder-decoder architecture.
   - Advantages: The usage of RVQ can largely improve the quality of discrete audio representations. And the usage of intermediate text translation can significantly improve the reliability of response contents.
   - Limitations: The multi-channel data structure requires the alignment between text sequences and audio sequences, which is a non-trivial engineering work. Also, the encoder-decoder RQtransformer architecture requires to receive the entire input of speaker’s channel, which still somehow downgrades the modeling efficency. Last but not least, this model can be regarded as alternative form of online cascading approach, which relies on the accuracy of both audio-to-text and text-to-audio generation.

In comparison to the above models, **Parrot** enjoys several important advantages:

- **Real Streaming Inference: Parrot** is capable of managing real-time streaming inference, eliminating the need for specific training on turn-taking, as required by models like LSLM. It can interact seamlessly with human users through natural turn-taking for the duration of the service. In contrast, multi-modal speech LLMs such as Mini-Omni and Llama-Omni can only interact with users on a turn-by-turn basis. In essence, **Parrot** does not depend on manually-defined interruption rules when conducting streaming inference.

- **Decoder-only Transformers:** In contrast to the encoder-decoder dialogue language modeling, **Parrot** employs a decoder-only transformer. This architecture offers numerous significant advantages. For instance, the encoder-decoder structure necessitates maintaining a window to receive complete inputs during the inference stage. However, the decoder-only architecture simply requires querying the cached key-value pairs, resulting in superior computational efficiency during inference.

- **Spoken Dialogue Data Usage Efficiency:** Both Moshi and LSLM randomly assign one channel as the speaker and another as the listener. This approach potentially reduces dialogue data efficiency, as the trained model becomes speaker-dependent. Essentially, the model needs to train the reverse conditional distribution by swapping the roles, which could pose scalability issues as more channels are added in the future. In contrast, **Parrot** is speaker-independent and concurrently learns the conditional distribution of both speaker's audio channels.

A.2 Representation of the joint sequence and mask strategy modeled by Parrot

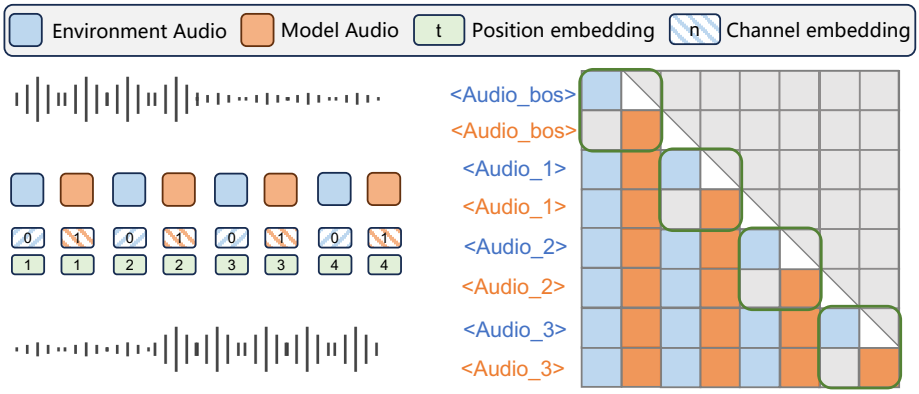


Figure 5: Pair-wise causal masking attention for next-token-pair prediction.

A.3 Potential Solutions to Limitations of Parrot

To overcome the limitations previously discussed, we propose a potential solution: the creation of a novel generative model for RVQ-based dual-channel audio sequences. However, the complexity of this task is heightened due to the unclear dependency relations across two distinct dimensions - the time dimension and the residual token dimension. As an alternative, we could opt to refine our method by increasing the number of discrete tokens per second. This approach would circumvent the need for RVQ while simultaneously enhancing the quality of the audio information. In future research, our goal is to train our method on substantially larger datasets and concurrently develop more sophisticated speech language model architectures. We hypothesize that the performance of our method can be further elevated to a new level through various potential approaches, without the direct application of RVQ.

A.4 More Implementation Details and Hyper-parameter Settings

A.4.1 Hyper-parameter Settings

Our model is trained on 16 A100 GPUs, utilizing a cosine annealing learning rate scheduler with a minimum learning rate of 4e-6 and a maximum learning rate of 4e-4. Each training epoch consists of 40,000 steps, with batch size 192 for each step. During fine-tuning, we use learn rate from 4e-6 to 5e-5.

A.4.2 Dataset

Parrot employs a two-stage training process. In the first stage, to establish foundational speech capabilities, we trained the model using three speech datasets totaling approximately 14,000 hours. This stage focuses on both speech understanding and synthesis. Unlike other models [6] that require audio to be transcribed into text, our **Parrot** only needs single-channel audio for direct training. This reduces the data requirements and, consequently, increases the amount of training data available. For the second stage, we need the **Parrot** to simultaneously gain the ability to listen and speak. To achieve this, we further utilize the Fisher dataset [3]. This dataset comprises 2200 hours of phone conversations between randomly paired participants, each discussing a given topic. A notable feature of the Fisher dataset is that each side of the conversation is recorded on separate channels, which allows us to provide ground-truth separated streams to **Parrot**. The original audio is sampled at 8kHz, and we use Librosa [1] to upsample it to 16kHz.

A.4.3 Baselines

We compare against baselines from the audio language modeling literature, in three settings. The first category encompasses audio-only models starting from a random initialization, including dGSLM[12]. The second category encompasses several newly released speech LLMs[23, 21, 6]. As a way to measure the impact of two stage training on spoken fluency, we compare these baselines with **Parrot** trained with and without pre-training phase.

Table 2: The datasets and their usage for training **Parrot**.

Type	Stages	Dataset	Hours
English Reading speech	1	LibriSpeech [15]	1,000 h
Pronunciation recording	1	Common Voice [1]	3,554 h
Video audio	1	Gigaspeech [2]	10,000 h
Spoken English audio	1	Libri-light [8]	60,000 h
Recorded telephone conversation	2	Fisher dataset [3]	2,000 h
Speech Instruction	2	InstructS2S-200K[6]	100 h

A.5 Training details

Large Language Model: In this study, we conceptualize audio as an additional language and employ three of the most widely recognized open-source LLMs as our foundational models: Llama-3.1-8B[5], Mistral-7B-v0.3[7], and Gemma-2-9B[19]. Each of these models comprises an embedding layer, multiple transformer blocks, and a language model (LM) head layer. They all encode the relative positional information of tokens using rotary positional encoding [17]. **Audio Tokenizer**: We train an audio tokenizer based on [20], which encodes each second of audio into 30-50 discrete tokens from a codebook of size 2048.

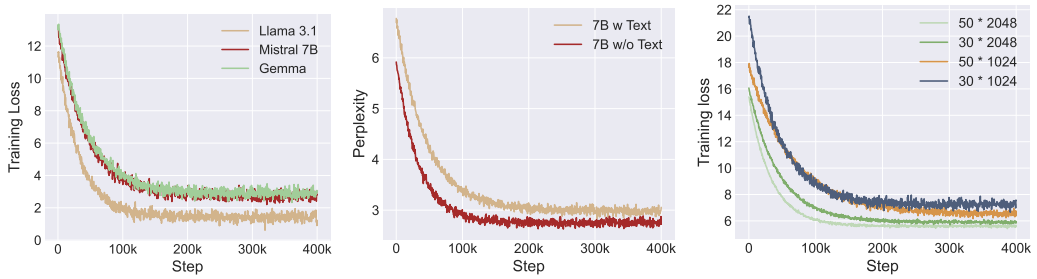
A.6 Pretrain Evaluation

Single audio channel language modeling: We begin by evaluating the capability of **Parrot** to model speech sequences through next-token prediction on the large-scale single channel audio dataset. We use perplexity on the test set's single-channel audio as the metric. The 6a presents the training loss over steps for three distinct models. All three models exhibit a decreasing trend in training loss, indicating effective learning over time. Mistral 7B and Gemma demonstrate similar training loss curves. Notably, Llama 3.1, which exhibits superior text reasoning capabilities, achieves a lower

[1] https://librosa.org/doc

training loss more rapidly compared to Mistral 7B and Gemma. This observation supports our hypothesis that stronger text models can be more effectively adapted to audio tasks, aligning with the conceptualization of "audio as a new language."

We also explore the trade-off between token rate and codebook size to optimize streaming interaction performance in Figure 6c. Notably, the configuration of 30 * 2048, which represents our chosen compromise solution, demonstrates a balanced performance with a steady decline in training loss.



(a) Training loss curve of different foundation models.

(b) The effectiveness of pre-training without text.

(c) The impact of token rate and codebook size.

Figure 6: Training loss and perplexity curves for **Parrot** under various Pretraining settings.

A.7 More Experimental Results

A.7.1 Audio tokenizer quality

Table 3: Comparison of different models and tokenizers on objective and subjective metrics.

Model	Tokenizer	Objective		Subjective	
		WER↓	SIM↑	MOS↑	SMOS↑
Groundtruth		1.9	0.93	4.5	3.96
VALL-E	EnCodec	7.9	0.75	3.08	3.31
USLM	SpeechTokenizer	7.2	0.81	3.63	3.45
Parrot	VQVAE	**6.9**	**0.82**	**3.71**	**4.50**

A.7.2 Dialogue linguistic quality

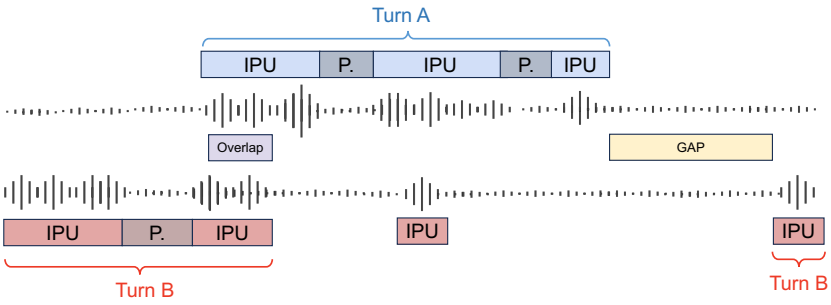


Figure 7: Illustration of turn-taking events: IPU (Interpausal Unit), Turn (for speaker A and Speaker B, resp), P.(within-speaker Pause), Gap and Overlap.

Our model generates two audio channels at the same time, allowing us to use basic Voice Activity Detection (VAD) tools on the output to gather turn-taking metrics. According to the settings in [12], an Inter-Pausal Unit (IPU) is a continuous speech segment within one speaker's channel, bordered by VAD-detected silences longer than 200ms on both ends. Silence is defined as the lack of voice signals

on either channel, while overlap refers to segments where voice signals are detected on both channels. Silences can be further divided into gaps (between IPUs of different speakers) and pauses (within the same speaker's IPUs). Consecutive IPUs by the same speaker, separated by a pause, are merged into a single turn. Our analysis will focus on measuring the duration distribution of IPUs, gaps, pauses, and overlaps in both the training corpus and the dialogues generated by our various models.

A.8 Motivations of Using Double-Channel Spoken Dialogue Data

Inspired by GPT-4o [14], we aspire to create a powerful voice assistant that can engage with human users in a natural and fluent way. Ideally, the assistant should be able to be interrupted by users. If a user needs to convey something urgently, the assistant should stop speaking and listen attentively. Furthermore, when a user is in thought or taking a pause, the assistant should not prematurely conclude that the user has finished speaking. Instead, it should patiently wait for the user to complete their thoughts. An advanced voice assistant could even interrupt users when it has already grasped their intentions, much like how we often interrupt each other in daily conversations. There are numerous other scenarios that an intelligent voice assistant should be equipped to handle. Given these complex application scenarios, it's challenging to address these issues through simple manual engineering, such as the introduction of special tokens like silence tokens, or hard interruptions when the user is speaking.

The success of foundational models hinges on our trust in the model's capacity to learn autonomously from data, rather than over-interfering with the learning process or over-engineering the neural architectures and algorithms. Consequently, in this paper, we utilize double-channel dialogue data and directly train the speech LLM on this spoken dialogue data. With robust pre-trained speech LLMs, we can reasonably anticipate that the model can learn how humans converse with each other by directly "reading" their dialogues. This approach eliminates the need for setting manual rules to assist the voice assistant in scenario judgement. The assistant may learn how to navigate these scenarios by processing a sufficient amount of spoken dialogue data. Regrettably, the current availability of open-source double-channel spoken-dialogue data is limited. Looking ahead, we hope our work will stimulate the community to gather large-scale double-channel or even multi-channel spoken dialogue data.

A.9 Reflective pause audio dataset

Prompt for reflective pause

"Hmm..., this question is a bit complicated, I need to think about it."
"Let me recall, uh..., yes, we went to the park that day."
"You know, that..., oh, yes, it's the new restaurant."
"I remember he mentioned it, um..., it seems to be last Friday."
"This matter, um..., I think we need to discuss it again."
"Let me think about it, uh..., yes, that's it."
"I'm not sure, um..., maybe I need to confirm it again."
"This question, um..., I think we can solve it this way."
"Let me think about it again, uh..., yes, I remember it."
"The one you mentioned, um..., I seem to have some impression."
"We need to deal with the budget issue of this project. Um..., this problem is a bit complicated, I need to think about it."
"Do you remember the last time we met? Let me recall, uh..., yes, we went to the park that day."
"Have you heard about the new restaurant? You know, that..., oh, yes, that new restaurant."
"When did he tell you the news? I remember he mentioned it, uh..., it seems to be last Friday."
"Do you have any suggestions about this plan? This matter, uh..., I think we need to discuss it again."
"Can you give me an example? Let me think about it, uh..., yes, that's it."
"Are you sure this data is correct? I'm not sure, uh..., I may need to confirm it again."
"How should we deal with this emergency? This problem, uh..., I think we can solve it this way."
"Can you explain this concept again? Let me think about it again, uh..., yes, I remember it."
"Do you know what he is talking about? The one you said, uh..., I seem to have some impression."

Prompt for GPT score

Content (1-5 points):
1 point: The response is largely irrelevant, incorrect, or fails to address the user's query. It may be off-topic or provide incorrect information.
2 points: The response is somewhat relevant but lacks accuracy or completeness. It may only partially answer the user's question or include extraneous information.
3 points: The response is relevant and mostly accurate, but it may lack conciseness or include unnecessary details that don't contribute to the main point.
4 points: The response is relevant, accurate, and concise, providing a clear answer to the user's question without unnecessary elaboration.
5 points: The response is exceptionally relevant, accurate, and to the point. It directly addresses the user's query in a highly effective and efficient manner, providing exactly the information needed.

Style (1-5 points):
1 point: The response is poorly suited for speech interaction, possibly including structured elements like lists or being overly complex, disjointed, or difficult to understand.
2 points: The response is somewhat suitable but may be too long, too short, or awkwardly phrased, making it less effective in a speech interaction context.
3 points: The response is generally suitable for speech interaction, but it may have minor issues with length, clarity, or fluency that detract slightly from the overall effectiveness.
4 points: The response is well-suited for speech interaction, with appropriate length, clear language, and a natural flow. It is easy to understand when spoken aloud.
5 points: The response is perfectly suited for speech interaction. It is the ideal length, highly clear, and flows naturally, making it easy to follow and understand when spoken.

Below are the transcription of user's instruction and models' response:
[Instruction]: **{instruction}**
[Response]: **{response}**

After evaluating, please output the scores in JSON format: {"content": content score, "style": style score}. You don't need to provide any explanations.