MAPACHE: MASKED PARALLEL TRANSFORMER FOR ADVANCED SPEECH EDITING AND SYNTHESIS

Guillermo Cámbara, Patrick Lumban Tobing, Mikolaj Babianski, Ravichander Vipperla, Duo Wang
Ron Shmelkin, Giuseppe Coccia, Orazio Angelini, Arnaud Joly, Mateusz Lajszczak, Vincent Pollet

Amazon Science

ABSTRACT

Recent advancements in Generative AI, such as scaled Transformer large language models (LLM) and diffusion decoders, have revolutionized speech synthesis. With speech encompassing the complexities of natural language and audio dimensionality, many recent models have relied on autoregressive modeling of quantized speech tokens. Such an approach limits speech synthesis to left-to-right generation, making these models unsuitable for speech edits free from audio discontinuities. We introduce Mapache, a novel architecture that combines a non-autoregressive masked speech language model with acoustic diffusion modeling, offering a unique, fully parallel pipeline. Mapache excels in precise speech editing that is indiscernible to human listeners, exhibiting inpainting and zero-shot synthesis capabilities that either surpass or rival those of other state-of-the-art models that specialize in just one of these tasks. This paper also sheds light on optimizing the decoding process for such non-autoregressive models.

Index Terms— text-to-speech, speech editing, transformers, diffusion, generative AI

1. INTRODUCTION

Latest innovations in Generative AI, such as scaling Transformer LLMs with data [1, 2] and using diffusion techniques [3] for text-conditioned image generation [4, 5], have greatly enhanced output quality. These methods have quickly been adapted to speech synthesis, setting new standards in machine-generated vocals and enabling zero-shot speech synthesis for unseen voices from minimal reference samples [6, 7, 8].

Speech encompasses the semantic complexity of natural language and the high dimensionality of audio. To handle this, recent methods discretize speech into token sequences using audio codecs [9, 10] that integrate with LLMs alongside text prompts. These speech tokens are then converted back to waveforms either by codec decoders or generative models like diffusion. For instance, Tortoise [6] uses a VQ-VAE for speech quantization and an autoregressive (AR) Transformer for token prediction, complemented by a diffusion model and a vocoder for waveform decoding. In contrast, VALL-E [11] uses EnCodec [10] for speech discretization and skips diffusion by utilizing EnCodec's decoder. AudioLM [7] predicts tokens in a hierarchical manner, focusing first on semantic tokens and then on SoundStream codec tokens [9].

The models previously mentioned produce expressive speech in zero-shot text-to-speech contexts but face design-related constraints. Specifically, AR LLMs inherently operate causally, restricting their ability to seamlessly edit existing speech. SoundStorm [12] and SpeechX [13] address this by using non-autoregressive (Non-AR) Transformers for acoustic tokens while retaining AR models in earlier stages. In contrast, NaturalSpeech2 [8] and Voicebox [14] fully parallelize processes, eliminating the LLM component and using diffusion and flow-matching [15] techniques, respectively.

In this study, we introduce a unique speech model combining a Masked Non-AR LLM Transformer with a diffusion decoder, named Mapache (raccoon in Nahuatl and Spanish, an animal with mask-like markings). This creates a fully parallel pipeline for smooth, high-quality speech editing. Mapache allows detailed speech adjustments, including repainting segments and inpainting from syllables to phrases with automatic text alignment. It can also generate expressive zero-shot text-to-speech from brief prompts (2-5 seconds). Our evaluations indicate that Mapache's edits are practically indistinguishable to human listeners and match or surpass the capabilities of other open-source state-of-the-art models; see publicly available samples here[1]. We also discuss refining the scheduled parallel decoding of such Non-AR models. Thus, the main contributions of this work are listed as:

I. Introducing Mapache - a parallel speech generation model based on non-autoregressive Transformers and diffusion.

II. Showing that Mapache seamlessly edits semantic content in speech and synthesizes speech from unseen speakers at a competitive level against open-source baselines, where these baselines can either do editing or synthesis only, but not both.

III. Pointing that speech may require augmenting the number of steps in scheduled parallel decoding for larger masked sequences, as opposed to fixed-size images.

2. MAPACHE

Mapache's cornerstone is a parallel sequence-to-sequence Transformer that models discrete text and speech representations, facilitating intricate speech editing and synthesis. However, Mapache comprises four modules: I) a speech tokenizer (VQ_{ENC}), II) the aforementioned parallel language-modeling Transformer (SpeechMPT), III) a diffusion decoder (DiT-DDPM), and IV) a vocoder (UnivNet). Initially, the tokenizer compresses speech signal S into a token sequence σ. Depending on the editing intent, parts of σ are masked for inpainting or appended for outpainting. This modified sequence, σ_m, combined with a text prompt θ, is processed by SpeechMPT, the core component of Mapache, producing speech tokens σ_θ and embeddings e_{σ_θ}, which integrate the semantic content from text θ. We upscale σ_θ or e_{σ_θ} to log-mel spectrograms μ_θ using a diffusion decoder and then synthesize waveform S_θ with a vocoder. A detailed Mapache structure is presented in Figure 1, with component specifics in subsequent sections.

[1] https://www.amazon.science/blog/generative-ai-enables-seamless-editing-of-recorded-speech

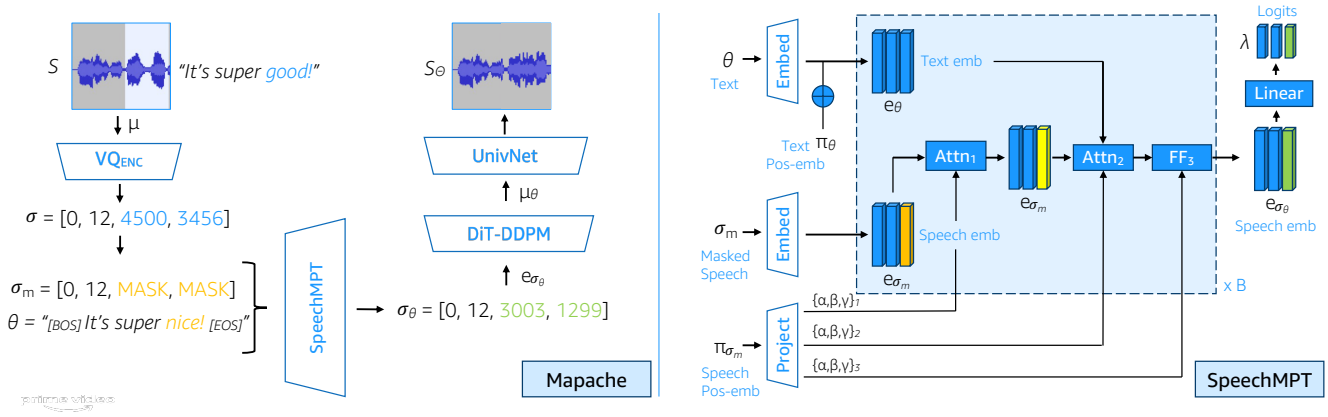

Fig. 1: (Left) **Mapache** architecture overview. (Right) **SpeechMPT** diagram for the forward pass.

2.1. Speech Tokenizer

To encode audio into a series of discrete speech tokens we employ a VQ-VAE[2] model [16] using a 512 hidden dimension, two symmetric CNN layers, 16 ResNet blocks, and an RNN prosody encoder post VQ codebook. We convert a waveform S sampled at 24 kHz into a log-mel spectrogram μ with 100 bins at 94 Hz. This spectrogram is tokenized into σ using the vector-quantized encoder VQ_{ENC}, resulting in a discrete space of $K = 8192$ tokens. The downsampling rate from the log mel-spectrogram is 4, making σ's rate equal to 23 Hz, closer to textual data resolution. The VQ-VAE's optimization involves an MSE loss between the decoder prediction y_μ and the actual μ, along with a commitment loss. We promote diverse codebook usage with a 16-dimension codebook and optimize using a learning rate of $3e^{-4}$ through AdamW.

2.2. Speech Masked Parallel Transformer

Speech, after being compressed by the VQ_{ENC}, is in a discrete form σ with lower entropy that is processed altogether with text by the Speech Masked Parallel Transformer (SpeechMPT), the key component of Mapache. SpeechMPT is inspired by MaskGIT [17] and Muse [18], which operate similarly in the image modality. First, we prepend beginning-of-sequence ($[BOS]$) and append end-of-sequence ($[EOS]$) tokens to a text prompt θ that contains the content to be pronounced. Then, the text is tokenized with a Byte-level Byte-Pair Encoding (BBPE) tokenizer of 2051 tokens to allow the model to be flexible for unseen characters. The text prompt is embedded through an embedding layer, yielding e_θ, which has dimension $D = 1024$, to which we add learnable positional embeddings π_θ. Regarding the speech prompt σ, we use a $MASK$ token to mask the parts of it that we want to generate, yielding σ_m. We embed it as well into e_{σ_m} with dimension D, and we extract the learnable positional embeddings π_{σ_m}. Now, we pass e_θ, e_{σ_m}, and π_{σ_m} through SpeechMPT's main stack of $B = 30$ Transformer blocks. Each block consists of two 16-head attention layers followed by a feed-forward layer (FF_3) of 2730 hidden dimension with GeGLU activations [19], each with skip connections. The first attention layer ($Attn_1$) does self-attention over e_{σ_m}, and the second one ($Attn_2$) does cross-attention between text and speech embeddings e_θ and e_{σ_m}. With respect to the speech positional embeddings π_{σ_m}, we propagate them through a SiLU activation [20] and a linear layer to regress the α and β affine parameters for each Layer Normalization (LN) within the attention and feedforward layers, plus a parameter γ that gates them. This conditioning technique, called Adaptive Layer Normalization (AdaLN), was inspired by the DiT text-to-image model [19] and proved to yield better alignment between text and output speech than classic sum during early explorations with SpeechMPT. The output from the Transformer blocks is a set of refined speech latent embeddings e_{σ_θ}, that we project into logits λ with a linear projection. The following steps vary depending on whether the model is in training or inference mode.

2.2.1. *Training*

In the training phase, we use the indices of the masked positions i to get their logits λ_i and the corresponding ground truth tokens from speech tokenized sequence σ_i, and we compute the cross-entropy loss between them, see Equation 1. The way we automatically mask tokens in training data is identical to Muse [18], as we use a Cosine schedule to sample a different masking percentage of the total sequence $r \in [0, 100]\%$ for each batch. This yields an expected 64% of masked tokens, often sampling high masking percentages. SpeechMPT is optimized for 440K steps using AdamW with an effective batch size of 64 samples. The learning rate warms up for 30K steps until it reaches $3e{-}4$, remains steady for 1K steps, and then decays for 300K steps to plateau at $5e{-}5$.

$$\mathcal{L}_{\text{CE}}(\sigma, \lambda) = -\sum_{j \in i} \sigma_j \log \left(\frac{e^{\lambda_j}}{\sum_{k=0}^{K-1} e^{\lambda_{j\,k}}} \right) \tag{1}$$

2.2.2. *Inference*

At inference, we compose the mask manually to edit specific parts of a speech prompt, or, in the case of a text-to-speech task, we predict the total duration of speech from the text prompt with a duration model (described in Section 2.4). To generate speech, we follow the scheduled parallel decoding scheme defined in MaskGIT [17], running T forward passes, removing the mask from the most confidently predicted tokens at each step while gradually annealing a starting sampling temperature of 5.0, encouraging a wider sampling in the early steps, which is narrowed down as decoding comes to its end.

[2] https://github.com/lucidrains/vector-quantize-pytorch

2.3. Diffusion Decoder & Vocoder

To transform the decoded speech tokens σ_θ into a log-mel spectrogram μ_θ, we employ a Denoising Diffusion Probabilistic Model[3] (DDPM) that is inspired by the text-to-image Diffusion Transformer (DiT) model [21], which we name DiT-DDPM. The diffusion process can be conditioned directly with the speech tokens σ_θ or with the latent embeddings e_{σ_θ} from SpeechMPT's last layer, which contains more semantic information than tokens. Thus, similarly to Tortoise [6], we first train DiT-DDPM with speech tokens, optimizing noise reconstruction MSE loss and ELBO over 4000 diffusion steps and a linear beta schedule, although we do only 40 steps at inference, enough for good perceptual quality. DiT-DDPM is trained for 480K steps using an effective batch size of 64 with a learning rate starting at $6e^{-5}$ that decays 98% every 20K steps. Once SpeechMPT has converged, we fine-tune DiT-DDPM to SpeechMPT latents for 200K more steps with effective batch size 128, which enhances the speaker identity and speech articulation in the output.

We convert speech tokens to 1024-dimensional embeddings with an embedding layer, and then we refine them with three layers of 16-head self-attention. For SpeechMPT latents, we apply a convolutional layer of kernel size 3 and stride 1, followed by four layers of self-attention. Additionally, we extract a conditioning latent from another sentence of the target speaker, which we use to scale and shift the speech embeddings to enrich their speaker identity information. Furthermore, we extract timestep embeddings by projecting sinusoidal embeddings with an MLP using SiLU activation. We then use a ResBlock followed by self-attention for three layers over the joint speech and timestep embeddings to integrate such features, and we use them for conditioning the diffusion process by applying the same AdaLN scheme from SpeechMPT and DiT [21]. This yields stronger conditioning, leading to observed faster and better convergence during early explorations of DiT-DDPM. Finally, we vocode output log-mel spectrograms with an open-sourced [4] UnivNet [22].

2.4. Duration Model

For doing text-to-speech from scratch, we use a duration model to predict speech output duration from text tokens. This model features a 6-layer Transformer encoder with 256-dim features and 8-head attention. Optimized over 100k steps using Huber loss, it regresses log-duration in seconds. This prediction determines the number of masked tokens for SpeechMPT to process.

3. EXPERIMENTS

We started evaluating Mapache with an initial editing task of reconstructing the content in masked speech segments (repainting), conditioning on ground truth transcripts. We found insights on particular best practices for speech parallel decoding, plus achieved positive results showing that repainted samples were indiscernible from revocoded speech for human listeners. This motivated us to run a second experiment with an increasing complexity of the task. Instead of repainting, we would inpaint new syllables, words, and sentences in speech recordings. Objective and subjective evaluations showed the proficiency of the model to perform such tasks even for larger sentences, which encouraged us to run a third experiment. This time, we would use a speech prompt to outpaint new utterances in it, thus performing zero-shot speech synthesis with unseen speakers. Find details on our experimental setup in the following subsections.

[3] https://github.com/openai/improved-diffusion

[4] https://github.com/maum-ai/univnet

Table 1: Training data composition.

Dataset	Hours	Speakers
LibriVox [23]	49K	10.5K
People's Speech [24]	2.7K	Unlabeled
Internet Archive Podcasts [5]	435	Unlabeled
Common Voice [25]	393	25.9K
VoxCeleb [26]	330	6.8K
English Accents [27]	16	28
Amazon In-house	1K	448

3.1. Datasets & metrics

All the model components were trained with a dataset composed of 54K hours of speech data, except the open-sourced UnivNet vocoder, which was not fine-tuned. The main bulk of the dataset consisted of public data, plus a $\sim 2\%$ of the total amount coming from Amazon's in-house data with multimedia and entertainment styles to enhance the prosody span in our database. See Table 1 for more information on data composition.

We used a validation set composed of 66 samples from 28 speakers, sampled from LibriVox, VoxCeleb, English Accents, and Amazon's in-house data, to cover a variety of speaking styles. We tuned with it the decoding timesteps as a function of masked tokens in the sequence. For the final evaluations, we split a test set of 25 samples from 13 speakers that were not seen at training. The test set contained only speakers from public data to have a fair comparison against open-source baselines. Pronunciation was assessed automatically by measuring the Word Error Rate (WER) with the open-sourced Whisper-Medium ASR[6], and for speaker similarity (SIM), we computed the cosine distance between synthetic and ground truth speech with speaker verification WavLM[7] embeddings.

3.2. Repainting

First, we assessed how the amount of masked speech tokens impacted quality, trying different numbers of decoding steps against our validation set. As can be observed in Figure 2, speaker similarity is affected by losing considerable speaker information as more speech is masked. However, correct pronunciation can be kept by doing more decoding steps as masking increases since the text prompt can always be consulted by the model, with 12-18 steps as a good range for consistent outputs. These findings suggest that the optimal number of decoding steps for speech might be dynamic, depending on the total length of the sequence to be generated, as opposed to fixed size image generation [17, 18].

To measure the acoustic loss related to the acoustic components on their own, we resynthesized ground truth recordings without any masking by running diffusion directly over unmasked SpeechMPT latents and then vocoding back to the waveform. As we can see in Table 2, the speaker similarity only dropped by 5% for the unseen test speakers, and the WER remained practically the same. Thus, we established this resynthesis pipeline as a baseline to assess the action of SpeechMPT on the semantic content separately.

For the test set, we used 18 decoding timesteps and a temperature of 5.0, a configuration that we kept for the rest of the experiments to ensure good quality and expressivity. Table 2 shows that with 20% masking, WER is close to ground truth recordings, and it

[5] https://archive.org/collection.php?identifier=podcasts

[6] https://huggingface.co/openai/whisper-medium

[7] https://huggingface.co/microsoft/wavlm-base-plus-sv

Table 2: Evaluation metrics in percentages (%) for repainting, inpainting and synthesis tasks.

	I - Repainting mask (%) = [20, 50, 70]			**II - Inpainting** [syllable, word, phrase]			**III - Synthesis**		
	WER ↓	**SIM** ↑	**PREF** ↑	**WER** ↓	**SIM** ↑	**MUSHRA** ↑	**WER** ↓	**SIM** ↑	**MUSHRA** ↑
Original	5.4	100	-	-	-	-	5.4	100	75.4 ± 0.7
Resynth	[5.3, 5.3, 5.3]	[**95**, **95**, **95**]	[**52**, **51**, **52**]	-	-	-	-	-	-
A3T	-	-	-	[9.8, 12.3, 21.0]	[90, 85, 74]	[62.1, 55.9, 45.7]	-	-	-
YourTTS	-	-	-	-	-	-	7.1	84	48.6 ± 0.8
Tortoise	-	-	-	[**2.9**, **4.0**, **3.3**]	[83, 82, 83]	[66.8, 66.6, 70.0]	**2.3**	83	**68.3** ± 0.7
Mapache	[**5.2**, **3.5**, **3.9**]	[94, 91, 89]	[**48**, **49**, **48**]	[5.5, 4.1, 4.0]	[**95**, **94**, **93**]	[**76.3**, **77.3**, **76.1**]	**2.2**	**87**	**67.8** ± 0.7

even improves with 50% and 70% masking. However, speaker similarity drops from 94% to 89% with more masking, as in validation.

We conducted a subjective preference test with 60 native speakers, comparing two speech samples: one resynthesizing ground truth speech with our acoustic components and the other a repainted version by Mapache. The test was done three times, varying the percentage of masked tokens (20%, 50%, and 70%). The results in Table 2 indicate a close preference gap, with $51-52\%$ for resynthesized and $49-48\%$ for repainted speech. Statistical analysis using p-values (0.43, 0.56, and 0.19) for the different masking levels showed no significance (compared to the 0.05 threshold), motivating us to inpaint new content into the original recordings in further experiments.

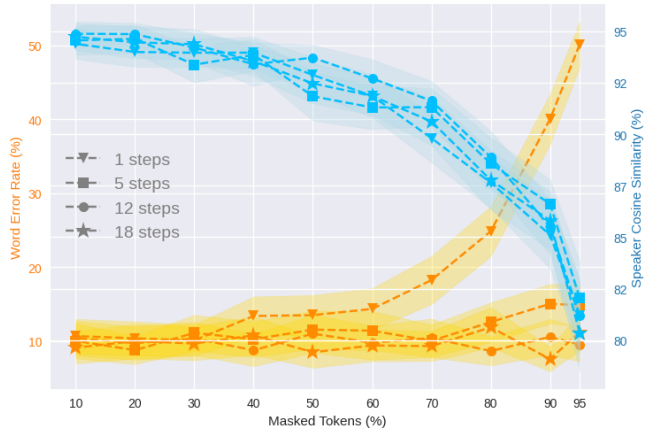

Fig. 2: Repainting performance in terms of WER (orange) and speaker cosine similarity (blue) for different mask percentages and decoding steps.

3.3. Inpainting

To continue with, we generated three other variants of the test set with Mapache, where we masked and inpainted new syllables, words, and phrases, respectively, within the original recording. We compared against A3T [28], as it is one of the few open-sourced speech editing baselines available. Note that A3T, as mentioned by its authors, is not a speech synthesis model, i.e., it cannot perform text-to-speech mechanism as ours. Plus, we also used Tortoise as a pure text-to-speech model baseline, where we regenerated the input recording by providing it as a reference to the model, and then changing the input text with the new syllable/word/phrase. We asked 67 participants in a MUSHRA test to score samples given how natural they sounded after listening to the original recording. The results in Table 2 show that Mapache got significantly higher scores compared to Tortoise and A3T. The latter's performance seems to decrease substantially when inpainting larger chunks like phrases, whereas Mapache shows consistent inpainting capabilities across different content edit lengths. This is also reflected in objective metrics, where speaker similarity is better preserved with Mapache, while WER scores of our model are comparable to recordings for syllable inpainting and even lower when editing words and phrases. Tortoise achieves best WER scores - we hypothesize that it synthesizes speech in the parts that are not inpainted that is more articulate than original recordings.

3.4. Zero-shot text-to-speech

Positive results in repainting and inpainting high chunks of speech motivated us to perform zero-shot text-to-speech. For that, we used a small speech prompt of 2 to 5 seconds, extracted from a random utterance of the target speaker similar to [12, 13] and extended it with a sequence of mask tokens with the duration predicted from text by our duration model. Then we used Mapache to generate speech over the masked region conditioned on target text. We synthesized the 25 samples in our test and compared against the original recordings and two publicly available state-of-the-art models that follow different paradigms: YourTTS (VITS-based) [29] and Tortoise (LLM-based). As seen in Table 2, Mapache came on par with Tortoise in terms of WER and significantly better in the objective speaker similarity score. Then, 81 native English speakers assigned MUSHRA scores to the systems in terms of how good speaker identity and style were recreated with respect to the reference samples. Mean MUSHRA scores were slightly higher for Tortoise, but a p-value of 0.55 suggests that the difference with Mapache is not significant, being both on-par, and YourTTS significantly behind with more mispronunciations and lower MUSHRA scores.

4. CONCLUSIONS

We have presented Mapache, a novel architecture that combines masked language modeling of discrete speech tokens and acoustic diffusion from latent speech, resulting in a fully parallel architecture that excels in speech editing workflows. It can regenerate parts of speech indistinguishable for human listeners and inpaint new content like words or phrases on existing speech with a high level of naturalness. Even more, it does zero-shot text-to-speech accurately, recreating unseen speaker identity and style based on a few seconds of audio reference, on-par with other publicly available state-of-the-art solutions. Opposed to the fixed-size text-to-image model, we hint that the number of decoding timesteps for these types of parallel models is variable depending on the amount of speech to generate. To improve speech editing capabilities, we foresee further work on adding more conditioning types (pitch curves, duration, style prompts), using LLM embeddings for text, and reinforcing acoustic diffusion with novel advances to get closer to lossless decoding.

5. REFERENCES

[1] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al., "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, pp. 9, 2019.

[2] Colin Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text Transformer," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[4] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen, "Hierarchical text-conditional image generation with clip latents," *arXiv preprint arXiv:2204.06125*, vol. 1, no. 2, pp. 3, 2022.

[5] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10684–10695.

[6] James Betker, "Better speech synthesis through scaling," *arXiv preprint arXiv:2305.07243*, 2023.

[7] Zalán Borsos et al., "AudioLM: a language modeling approach to audio generation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[8] Kai Shen et al., "NaturalSpeech 2: Latent diffusion models are natural and zero-shot speech and singing synthesizers," *arXiv preprint arXiv:2304.09116*, 2023.

[9] Neil Zeghidour et al., "SoundStream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.

[10] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[11] Chengyi Wang et al., "Neural codec language models are zero-shot text to speech synthesizers," *arXiv preprint arXiv:2301.02111*, 2023.

[12] Zalán Borsos et al., "SoundStorm: Efficient parallel audio generation," *arXiv preprint arXiv:2305.09636*, 2023.

[13] Xiaofei Wang et al., "SpeechX: Neural codec language model as a versatile speech Transformer," *arXiv preprint arXiv:2308.06873*, 2023.

[14] Matthew Le et al., "Voicebox: Text-guided multilingual universal speech generation at scale," *arXiv preprint arXiv:2306.15687*, 2023.

[15] Yaron Lipman et al., "Flow matching for generative modeling," *arXiv preprint arXiv:2210.02747*, 2022.

[16] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu, "Neural discrete representation learning," 2018.

[17] Huiwen Chang et al., "MaskGIT: Masked generative image Transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 11315–11325.

[18] Huiwen Chang et al., "Muse: Text-to-image generation via masked generative Transformers," *arXiv preprint arXiv:2301.00704*, 2023.

[19] Noam Shazeer, "GLU variants improve Transformer," *arXiv preprint arXiv:2002.05202*, 2020.

[20] Stefan Elfwing, Eiji Uchibe, and Kenji Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural networks*, vol. 107, pp. 3–11, 2018.

[21] William Peebles and Saining Xie, "Scalable diffusion models with Transformers," *arXiv preprint arXiv:2212.09748*, 2022.

[22] Won Jang, Dan Lim, Jaesam Yoon, Bongwan Kim, and Juntae Kim, "Univnet: A neural vocoder with multi-resolution spectrogram discriminators for high-fidelity waveform generation," *arXiv preprint arXiv:2106.07889*, 2021.

[23] Jodi Kearns, "LibriVox: Free public domain audiobooks," *Reference Reviews*, vol. 28, no. 1, pp. 7–8, 2014.

[24] Daniel Galvez et al., "The people's speech: A large-scale diverse english speech recognition dataset for commercial usage," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[25] Rosana Ardila et al., "Common Voice: A massively-multilingual speech corpus," in *Proceedings of the Twelfth Language Resources and Evaluation Conference*, 2020, pp. 4218–4222.

[26] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman, "VoxCeleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, vol. 60, pp. 101027, 2020.

[27] Isin Demirsahin, Oddur Kjartansson, Alexander Gutkin, and Clara Rivera, "Open-source Multi-speaker Corpora of the English Accents in the British Isles," in *Proceedings of The 12th Language Resources and Evaluation Conference (LREC)*, Marseille, France, May 2020, pp. 6532–6541, European Language Resources Association (ELRA).

[28] He Bai, Renjie Zheng, Junkun Chen, Mingbo Ma, Xintong Li, and Liang Huang, "A^3T: Alignment-aware acoustic and text pretraining for speech synthesis and editing," in *International Conference on Machine Learning*. PMLR, 2022, pp. 1399–1411.

[29] Edresson Casanova et al., "YourTTS: Towards zero-shot multi-speaker TTS and zero-shot voice conversion for everyone," in *International Conference on Machine Learning*. PMLR, 2022, pp. 2709–2720.