# SegINR: Segment-Wise Implicit Neural Representation for Sequence Alignment in Neural Text-to-Speech

Minchan Kim, *Graduate Student Member, IEEE*, Myeonghun Jeong, *Graduate Student Member, IEEE*, Joun Yeop Lee, and Nam Soo Kim, *Senior Member, IEEE*

***Abstract*—We present SegINR, a novel approach to neural Text-to-Speech (TTS) that eliminates the need for either an auxiliary duration predictor or autoregressive (AR) sequence modeling for alignment. SegINR simplifies the TTS process by directly converting text sequences into frame-level features. Encoded text embeddings are transformed into segments of frame-level features with length regulation using a conditional implicit neural representation (INR). This method, termed Segment-wise INR (SegINR), captures temporal dynamics within each segment while autonomously defining segment boundaries, resulting in lower computational costs. Integrated into a two-stage TTS framework, SegINR is employed for semantic token prediction. Experiments in zero-shot adaptive TTS scenarios show that SegINR outperforms conventional methods in speech quality with computational efficiency.**

***Index Terms*—Implicit neural representation, sequence alignment, text-to-speech.**

## I. Introduction

Neural Text-to-Speech (TTS) models inherently address the alignment problem by regulating sequence length, expanding text length into speech length based on the irregular monotonic alignment between text and speech. This alignment problem is typically tackled using intermediate frame-level features (e.g., mel-spectrogram, semantic tokens [1], [2], acoustic tokens [3], [4]) rather than raw waveforms. Conventional TTS models can be categorized into two types depending on alignment modeling: autoregressive (AR) and duration-based non-autoregressive (NAR) methods. AR models extend frames sequentially, dynamically determining the relevant parts of the text features, including attention-based sequence-to-sequence (seq2seq) models [1], [5], [6] and transducers [7], [8], [9]. However, AR models have drawbacks such as requiring recurrency during inference, which leads to slow inference and error propagation, especially with misalignment [10]. In contrast, duration-based NAR models [11], [12], [13] utilize explicit phoneme durations for length regulation, expanding text embedding sequences to align with frame-level features based on duration, then converting them into frame-level features in parallel using various generative models. These models rely on ground truth alignment acquired from forced alignment algorithms [14], [15], [16] during training and predicted durations obtained from an auxiliary duration predictor during inference, which induces the load of modeling a duration predictor.

In this paper, we propose a novel method that converts text sequences into frame-level features without requiring either an auxiliary duration predictor or AR sequence modeling. We assume that each frame in the encoded text embedding sequence can contain sufficient information for the corresponding segment of frame-level features. Following this assumption, we decompose the seq2seq task into a set of embedding-to-segment (emb2seg) conversions, which transform a text embedding into a segment of frame-level features. We build each emb2seg conversion model based on implicit neural representation (INR) [17], [18], [19], [20]. INR is a multi-layer perceptron (MLP) model that represents continuous signals as a function of coordinates. We construct a conditional INR that takes the time index $i$ within the segment as input and returns the $i_{th}$ frame of the segment, using the text embedding as a conditioning factor. This conditional INR, named Segment-wise INR (SegINR), represents the temporal dynamics of the frame-level feature within a segment assigned to each text unit. SegINR replaces length-expanded sequence modeling with building a function space of time. Additionally, we introduce an end of segment token $\varnothing$, allowing INR to automatically determine its own duration. By jointly predicting the output sequence and the $\varnothing$ token, the model can determine segment boundaries autonomously without using an external duration predictor. SegINR significantly reduces the computational cost of length-extended sequence modeling, as the proposed method only requires text-level sequence encoding and shallow MLP layers without receptive field. The final output sequence is a concatenation of all segments generated independently by the SegINR.

We explore the application of SegINR within a two-stage TTS framework in [2], which separates the TTS process into

Received 1 December 2024; revised 7 January 2025; accepted 8 January 2025. Date of publication 15 January 2025; date of current version 23 January 2025. This work was supported by Samsung Research, Samsung Electronics Company Ltd. The associate editor coordinating the review of this article and approving it for publication was Dr. Aykut Ko. *(Corresponding author: Nam Soo Kim.)*

Minchan Kim, Myeonghun Jeong, and Nam Soo Kim are with the Department of Electrical and Computer Engineering, Seoul National University, Seoul 08826, South Korea, and also with the Institute of New Media and Communications, Seoul National University, Seoul 08826, South Korea (e-mail: mckim@hi.snu.ac.kr; mhjeong@hi.snu.ac.kr; nkim@snu.ac.kr).

Joun Yeop Lee is with the Samsung Research, Seoul 06765, South Korea (e-mail: jounyeop.lee@samsung.com).

Digital Object Identifier 10.1109/LSP.2025.3528858

text-to-semantic token prediction and speech generation using semantic tokens. As alignment modeling is tackled in the first stage, we adopt SegINR for semantic token prediction. By targeting semantic tokens that encapsulate disentangled linguistic information, instead of directly modeling continuous speech features, we mitigate the inevitable discontinuities at segment boundaries. We then generate waveforms from semantic tokens using a masked language model from [21]. Our experiments in a zero-shot adaptive TTS scenario demonstrate that the proposed approach outperforms conventional methods. Generated samples are available on the demo page.[1]

## II. Backgrounds

### A. Length Regulation in TTS

*1) Attention-Based AR Models [1], [5], [6], [22]:* Attention-based AR models calculate alignment using an attention mechanism [23], [24], rather than defining explicit durations. This eliminates the need to calculate durations during both training and inference, simplifying the framework. However, their autoregressive nature leads to slow inference, and the attention mechanism can cause alignment failures [10], as it does not guarantee monotonic constraints.

*2) Transducer [7], [8], [9]:* Transducers are well-suited for seq2seq tasks with monotonic alignment, such as speech recognition [25], and have also been applied to TTS [7], [8], [9]. They construct an alignment lattice and define the conditional likelihood as the marginalization over all possible paths, using a special blank token $\varnothing$ to indicate transitions to the next input frame. However, like attention-based models, transducers operate autoregressively, leading to slow inference.

*3) Duration-Based NAR Models [11], [12], [13]:* Duration-based NAR models explicitly define durations for text units. Text embeddings are duplicated based on their durations and decoded to generate the output sequence, enabling parallel generation and fast inference. However, most of these models rely on external duration information from forced alignment algorithms [14], [15], [16] during training and require a duration predictor during inference. This introduces the additional complexity of building a duration predictor and cascading errors due to training-inference mismatch. Recent works [26], [27] address this mismatch with differentiable alignment.

### B. Implicit Neural Representation (INR)

Implicit Neural Representations (INRs) are neural networks that parameterize fields in continuous coordinates, enabling the representation of complex, high-dimensional data with a small number of learnable parameters. For example, a colored 2D image can be modeled as $R^2 \rightarrow R^3$, mapping pixel coordinates to RGB values. INRs are widely used in data compression [28], [29], 3D rendering [18], [19], [30], and generative modeling [20], [30], [31], [32], with advances in representing fine-grained details [17], [33]. In the audio and speech domain, [34], [35] have utilized conditional INRs and hypernetworks [36] for waveform generation. Unlike our approach, their focus is on modeling high-resolution waveforms rather than frame-level features considering seq2seq alignment.

INRs are well-suited for modeling speech features in TTS due to the inherent temporal continuity of speech. They can replace conventional sequence models, eliminating the need for recursive computations or large architectures like transformers, and instead rely solely on simple MLP layers to capture temporal dynamics. However, directly applying INRs in TTS is challenging for conventional conditional INRs due to seq2seq alignment and varying durations. To address this, we propose a segment-level approach that resolves alignment issues while leveraging the benefits of INRs.

## III. Method

### A. Segment-Wise Implicit Neural Representation (SegINR)

Given an input text sequence $\mathbf{x}_{1:U}$ and the corresponding frame-level features $\mathbf{y}_{1:T}$, our objective is to construct a model that converts $\mathbf{x}_{1:U}$ into $\mathbf{y}_{1:T}$. Adhering to the monotonic alignment constraint between text and speech, we define the duration as $\mathbf{d}_{1:U} \subseteq \mathbb{Z}_{\geq 0}$, where $\sum_{u=1}^{U} d_u = T$. Each input text token $x_u$ is aligned to the segment $\mathbf{y}^u_{1:d_u}$, which is a slice of $\mathbf{y}_{1:T}$ starting at index $\sum_{k=1}^{u-1} d_k + 1$, and ending at $\sum_{k=1}^{u} d_k$. Consequently, $\mathbf{y}_{1:T}$ is represented as the concatenation of all segments: $\mathbf{y}_{1:T} = \mathbf{y}^1_{1:d_1} \mid \mathbf{y}^2_{1:d_2} \mid \ldots \mid \mathbf{y}^U_{1:d_U}$.

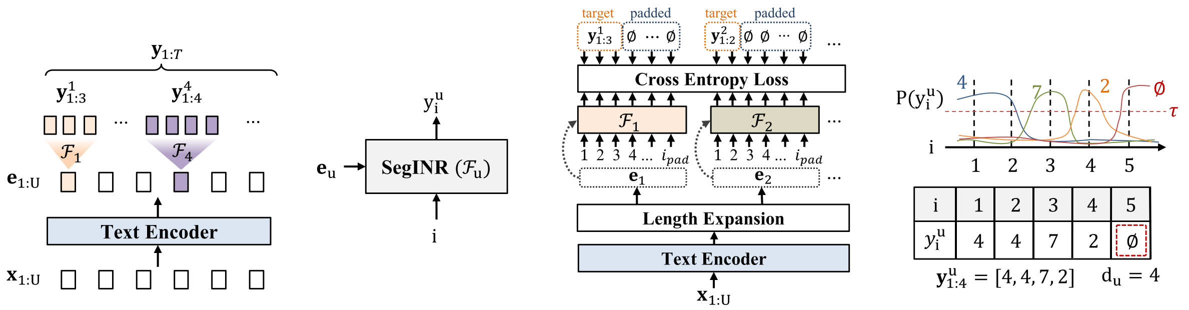
Regarding the text embedding sequence $\mathbf{e}_{1:U}$ obtained from a text encoder, we assume that each $\mathbf{e}_u$ can contain sufficient information for generating $\mathbf{y}^u_{1:d_u}$. Then, we breakdown the seq2seq problem into a set of embedding-to-segment (emb2seg) problems: generating $\mathbf{y}^u_{1:d_u}$ from $\mathbf{e}_u$, which also determines $d_u$ by itself.

We address the emb2seg problem using a conditional INR named SegINR. SegINR defines a function of the time index $i$: $\mathcal{F}_u(i; \mathbf{e}_u, \theta) = \mathbf{y}^u_i$, where $i \in \mathbb{R}$ such that $1 \leq i \leq d_u$ and $\theta$ indicates the parameters of SegINR. $\mathcal{F}_u$ leverages the inherent continuity of sequences to efficiently represent the temporal dynamics of each segment. Notably, the time index $i$ is treated as a real-valued scalar, even though only integer values are used in our framework. To automatically determine the domain $[1, d_u]$, we draw inspiration from the transducer framework and allow $\mathcal{F}_u$ to predict a special token $\varnothing$, which signifies the end of a segment. We set $\mathcal{F}_u(d_u + 1; \mathbf{e}_u, \theta) = \varnothing$. Consequently, $d_u$ is determined as the largest index before predicting $\varnothing$. After generating all segments independently for each $\mathbf{e}_u$, the entire sequence is constructed by concatenating the generated segments. The proposed framework is illustrated in Fig. 1(a) and (b).

### B. Application

*1) Semantic Token Prediction:* We integrate SegINR into a two-stage TTS framework in [2] comprising text-to-semantic token and semantic-to-acoustic token stages. In [2], the first stage employs a transducer, while the second stage uses a masked language model: G-MLM [21], separating coarse-grained linguistic modeling from fine-grained acoustic modeling. Since alignment is handled in the first stage, we replace the transducer with

[1] [Online]. Available: https://gannnn123.github.io/seginr



(a) Entire system (b) SegINR (c) Training (d) Inference

Fig. 1. Illustration of SegINR and its application for semantic token prediction: (a) overall concept of SegINR, (b) structure of SegINR, (c) training method for semantic token prediction, (d) inference method for semantic token prediction.

SegINR. Semantic tokens, used as target frame-level features, encapsulate linguistic and coarse-grained information, minimizing discontinuities at segment boundaries. Their discrete nature also simplifies the integration of the ∅ token into the output space by adding a single class to the categorical distribution, unlike continuous speech features.

*2) Architecture:* The entire semantic token prediction model consists of a text encoder and a SegINR. The text encoder is built using conformer blocks [37], while SegINR utilizes a modulated SIREN structure inspired by Coin++ [29]. The modulated SIREN comprises MLP layers with sine activation, modulated by a conditioning embedding $\mathbf{e}_u$.

*3) Training:* We jointly train the text encoder and SegINR using a single training loss. To calculate the ground truth duration, we utilize Token Transducer++ [2], a transducer designed for text-to-semantic token translation. The most probable path in the alignment lattice of the transducer is computed using the Viterbi algorithm, summing up the number of frames assigned to each phoneme.

We not only enforce the condition $\mathcal{F}_u(d_u + 1; \mathbf{e}_u, \theta) = \varnothing$, but also train with an auxiliary condition $\mathcal{F}_u(i; \mathbf{e}_u, \theta) = \varnothing$ for indices $i$ satisfying $d_u + 1 < i < i_{pad}$, where $i_{pad}$ is a constant for a sufficiently large padding number for ∅. This auxiliary training ensures the consistent output of ∅ for $i > d_u + 1$, thereby improving the stability of the inference process.

Although SegINR is not a sequence model but rather consists of MLP layers, we train it in a pseudo-sequential manner for convenience, which is implemented at the batch level, as illustrated in Fig. 1(c). After extracting $\mathbf{e}_{1:U}$, we expand each $\mathbf{e}_u$ $i_{pad}$ times and create a pseudo index sequence. We then feed both the pseudo index sequence and the expanded text embeddings. The entire model is trained using cross-entropy loss between the pseudo output sequence and the target sequence, which is the concatenation of all ∅-padded semantic token segments.

*4) Inference:* Once we obtain $\mathbf{e}_{1:U}$ from the text encoder, we generate $\mathbf{y}_{1:T}$ using SegINR, as illustrated in Fig. 1(d). During inference, SegINR returns the most probable semantic token at each time index if the estimated probability of ∅ is below a threshold $\tau$; otherwise, it returns ∅. A key advantage of SegINR is its compatibility with both streaming and parallel inference frameworks. 1) In the streaming scenario, we sequentially decode $\mathcal{F}_u$ by incrementing $i$ until ∅ is returned, then move on to the next $\mathcal{F}_{u+1}$ until $U$ is reached. This process is similar to the inference of transducers, but operates without recurrence. 2) For parallel decoding, we define $i_{\max}$, the maximum duration per text unit. All outputs are generated in parallel by injecting $[0, 1, 2, \ldots, i_{\max}]$ for each $\mathcal{F}_u$ in a batch process. We select only valid outputs, stopping at the first ∅. If ∅ is not returned by $i_{\max}$, we set $d_u = i_{\max}$. Although this method incurs some wasted computation due to abandoned outputs, SegINR's low computational cost results in significantly faster inference compared to other sequence-level decoding methods.

## IV. Experiments

### A. Experimental Setting

We conducted experiments on zero-shot adaptive TTS, following the experimental settings of previous work [2]. We used the same semantic tokens; the indices of $k$-means clustering on the wav2vec2.0-XLSR model [38] with $k = 512$. We built the semantic token prediction model using SegINR, replacing the Token Transducer++ in [2]. The proposed model was trained on all training subsets of the LibriTTS corpus [39] and evaluated on test subsets.

*1) Implementation Details:* For semantic token prediction, we used the same text encoder structure of the Token Transducer++ which consist of a conformer blocks [37]. The dimension of the text embedding $\mathbf{e}_u$ is 384. For SegINR, we implemented a modulated SIREN with three layers of MLP, each with a hidden dimension of 256 using the official code from Coin++ [29].[2] Through our experiments, we found that the SIREN activation frequency of $w_0 = 1.0$ performed well. For zero-shot adaptation, we added a reference encoder with the same structure as described in [2]. During training, the reference encoder processes a randomly cropped 3-second segment of the target speech as in [2]. The resulting reference embedding is then globally added at the beginning of the text encoder to condition prosody information. Also, we set $i_{pad} = 20$ for training SegINR, and $i_{\max} = 20$ and $\tau = 0.5$ for parallel inference. We trained the proposed model for 50 epochs with dynamic batch size containing up to 240 seconds.

*2) Baselines:* We used three baseline models for performance comparison: VITS [12] representing an NAR model,

[2] [Online]. Available: https://github.com/EmilienDupont/coinpp

VALLE-X [40] representing an AR model, and the model proposed by Lee et al. [2]. To adapt the baseline VITS to the zero-shot scenario, we incorporated the same reference encoder structure as in our proposed model. For VALLE-X, we used the open-source implementation.[3] The model by Lee et al. [2] served as the primary baseline in our work, as we shared the same semantic-to-acoustic token stage but differing in the semantic token prediction models: the Token Transducer++.

### B. Results: Zero-Shot Adaptive TTS

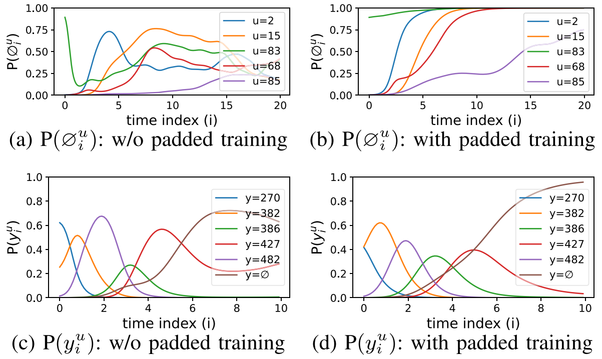
We evaluated the mean opinion score (MOS), similarity MOS (SMOS), character error rate (CER), speaker embedding cosine similarity (SECS), and real-time factor (RTF). MOS, rated on a 1–5 scale by 14 testers, measured perceptual speech quality. SMOS evaluated speaker similarity, with the same testers rating whether the synthesized and reference samples matched in timbre and prosody. CER, assessing intelligibility, was calculated using the Whisper large model [41]. SECS measured speaker similarity between synthesized and reference speech using the pre-trained WavLM large speaker verification model.[4] For objective evaluations, we randomly selected 800 text-reference speech pairs from the test set.

The proposed model outperformed baseline models across most subjective metrics. A t-test revealed significant differences among the baselines, except for the model by Lee et al. [2], in SMOS. Notably, SMOS, which evaluates both timbre and prosody similarity, sometimes resulted in higher scores for synthesized samples compared to ground truth, as the latter only guarantees speaker identity. VALLE-X had the highest CER due to misalignment issues common in attention-based AR models. Since SECS is largely influenced by the semantic-to-acoustic token conversion rather than semantic token generation [2], the model by Lee et al. [2] achieved scores similar to ours, as both use the same post-processing model. In terms of inference speed, VITS demonstrated the highest RTF. However, when comparing only the semantic token prediction part between the model by Lee et al. [2] and the proposed SegINR, the Token Transducer++ in [2] showed an RTF of 22.35, while SegINR achieved **134.85**. This substantial difference indicates the feasibility and computational efficiency of SegINR for sequence alignment, even though the other part, G-MLM, occupies a significant portion in our current framework.

### C. Ablation: Training and Inference Schemes

We analyzed SegINR's training and inference schemes. For training, we compared models trained to predict $\varnothing$ only at $i = d_u + 1$ (Fig. 2(a), (c)) with those using auxiliary padded $\varnothing$ (Fig. 2(b), (d)). For inference, we compared two methods: returning $\varnothing$ when its probability exceeds thresholds ($\tau = 0.2, 0.5, 0.8$) or when it has the highest probability among all candidates. Results are shown in Fig. 2 and Table II.

As shown in Fig. 2, the non-padded training model showed probabilities for $\varnothing$ that were not monotonically increasing, whereas the padded training model produced a monotonic increase in $\varnothing$'s probability due to the extrapolated constraints. Table II presents CER and duration ratio (the total duration of generated speech divided by the ground truth in the testset). Higher thresholds led to slower speech rates due to delayed $\varnothing$ emission. Padded training improved intelligibility, and $\tau = 0.8$ achieved the lowest CER but produced overly long durations, mismatching the ground truth. Without padded training, the model often failed to emit $\varnothing$, reaching $i_{\max}$ and causing high CER and exaggerated durations. We selected $\tau = 0.5$ as the default, balancing intelligibility and duration alignment. Notably, adjusting $\tau$ enables control over SegINR's speech rate.



(a) P($\varnothing_i^u$): w/o padded training (b) P($\varnothing_i^u$): with padded training

(c) P($y_i^u$): w/o padded training (d) P($y_i^u$): with padded training

Fig. 2. Comparison of the adoption of padded training: (a) and (b) show the probability of $\varnothing$, while (c) and (d) show the probability of $y$ for a fixed $u$.

TABLE I
RESULTS OF ZERO-SHOT ADAPTIVE TTS. MOS AND SMOS ARE REPRESENTED WITH 95% CONFIDENCE INTERVALS. RTF IS CALCULATED BY A QUADRO RTX8000 GPU

| **Method** | **MOS** | **SMOS** | **CER**(%) | **SECS** | **RTF** |
|---|---|---|---|---|---|
| Ground Truth | 4.51±0.07 | 4.38±0.08 | 1.56 | 0.678 | - |
| VITS [12] | 3.60±0.07 | 4.04±0.08 | 5.80 | 0.375 | **70.12** |
| VALLE-X [40] | 3.71±0.09 | 4.17±0.08 | 10.58 | 0.426 | 1.89 |
| Lee et al. [2] | 4.11±0.09 | 4.44±0.07 | 3.55 | **0.463** | 6.76 |
| Proposed | **4.27±0.08** | **4.51±0.07** | **3.14** | 0.461 | 8.72 |

TABLE II
CHARACTER ERROR RATES (CER) AND DURATION RATIOS FOR EACH CASE OF PADDED TRAINING

| **CER**(%) (duration ratio) | w/o padded training | with padded training |
|---|---|---|
| infer w/o $\tau$ | 8.55 (1.005) | 7.66 (0.783) |
| infer with $\tau = 0.2$ | 8.31 (0.823) | 10.68 (0.675) |
| infer with $\tau = 0.5$ | 8.30 (1.673) | 3.14 (0.995) |
| infer with $\tau = 0.8$ | 36.83 (3.905) | **2.49** (1.377) |

## V. CONCLUSION

We proposed SegINR, a novel framework for sequence alignment in TTS. By leveraging the concept of conditional INRs, we modeled frame-level speech features on a segment-wise basis and applied it to semantic token prediction tasks. Our results demonstrate the feasibility and superiority of SegINR. In future work, we plan to explore the use of SegINR for other speech features and to integrate it with various generative models to enhance its generative capabilities.

[3] [Online]. Available: https://github.com/Plachtaa/VALL-E-X

[4] [Online]. Available: https://github.com/microsoft/UniSpeech/tree/main/downstreams/speaker_verification

## REFERENCES

[1] E. Kharitonov et al., “Speak, read and prompt: High-fidelity text-to-speech with minimal supervision,” *Trans. Assoc. Comput. Linguistics*, vol. 11, pp. 1703–1718, 2023.

[2] J. Yeop et al., “High fidelity text-to-speech via discrete tokens using token transducer and group masked language model,” in *Proc. Interspeech*, 2024, pp. 3445–3449.

[3] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, “High fidelity neural audio compression,” *Trans. Mach. Learn. Res.*, 2023.

[4] D. Yang et al., “Hifi-codec: Group-residual vector quantization for high fidelity audio codec,” 2023, *arXiv:2305.02765*.

[5] J. Shen et al., “Natural TTS synthesis by conditioning wavenet on MEL spectrogram predictions,” in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2018, pp. 4779–4783.

[6] C. Wang et al., “Neural codec language models are zero-shot text to speech synthesizers,” 2023, *arXiv:2301.02111*.

[7] M. Kim et al., “Transduce and speak: Neural transducer for text-to-speech with semantic token prediction,” in *Proc. 2023 IEEE Autom. Speech Recognit. Understanding Workshop*, 2023, pp. 1–7.

[8] J. Chen et al., “Speech-T: Transducer for text to speech and beyond,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, vol. 34, pp. 6621–6633 .

[9] C. Du et al., “Vall-T: Decoder-only generative transducer for robust and decoding-controllable text-to-speech,” 2024, *arXiv:2401.14321*.

[10] C. Valentini-Botinhao and S. King, “Detection and analysis of attention errors in sequence-to-sequence text-to-speech,” in *Proc. Interspeech 2021, 22nd Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 2746–2750.

[11] Y. Ren et al., “Fastspeech 2: Fast and high-quality end-to-end text to speech,” in *Proc. Int. Conf. Learn. Representations*, 2021.

[12] J. Kim, J. Kong, and J. Son, “Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech,” in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 5530–5540.

[13] V. Popov, I. Vovk, V. Gogoryan, T. Sadekova, and M. Kudinov, “Grad-TTs: A diffusion probabilistic model for text-to-speech,” in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8599–8608.

[14] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, “Montreal forced aligner: Trainable text-speech alignment using kaldi,” in *Proc. Interspeech*, 2017, pp. 498–502.

[15] J. Kim, S. Kim, J. Kong, and S. Yoon, “Glow-TTs: A generative flow for text-to-speech via monotonic alignment search,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2020,vol. 33, pp. 8067–8077 .

[16] R. Badlani, A. Łańcucki, K. J. Shih, R. Valle, W. Ping, and B. Catanzaro, “One TTs alignment to rule them all,” in *Proc. 2022-2022 IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2022, pp. 6092–6096.

[17] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein, “Implicit neural representations with periodic activation functions,” in *Proc. Adv. Neural Inf. Process Syst.*, 2020, vol. 33, pp. 7462–7473 .

[18] B. Mildenhall et al., “NERF: Representing scenes as neural radiance fields for view synthesis,” *Commun. ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[19] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “DeepSDF: Learning continuous signed distance functions for shape representation,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 165–174.

[20] E. Dupont, H. Kim, S. M. Ali Eslami, D. J. Rezende, and D. Rosenbaum, “From data to functa: Your data point is a function and you can treat it like one,” in *Proc. 39th Int. Conf. Mach. Learn.,*, 2022, pp. 5694–5725.

[21] M. Jeong, M. Kim, J. Y. Lee, and N. S. Kim, “Efficient parallel audio generation using group masked language modeling,” *IEEE Signal Process. Lett.*, vol. 31, pp. 979–983, 2024.

[22] N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu, “Neural speech synthesis with transformer network,” in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 6706–6713,.

[23] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *Proc. 3rd Int. Conf. Learn. Representations*, Y. Bengio and Y. LeCun, Eds., San Diego, CA, USA, May 2015.

[24] D. Soydaner, “Attention mechanism in neural networks: Where it comes and where it goes,” *Neural Comput. Appl.*, vol. 34, no. 16, pp. 13371–13385, 2022.

[25] A. Graves, “Sequence transduction with recurrent neural networks,” in *Proc. Representation Learn. Worksop*, 2012.

[26] B. Nguyen, F. Cardinaux, and S. Uhlich, “Autotts: End-to-end text-to-speech synthesis through differentiable duration modeling,” in *Proc. 2023-2023 IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2023, pp. 1–5.

[27] J. Donahue, S. Dieleman, M. Binkowski, E. Elsen, and K. Simonyan, “End-to-end adversarial text-to-speech,” in *Proc. Int. Conf. Learn. Representations*, 2021.

[28] E. Dupont, A. Goliński, M. Alizadeh, Y. W. Teh, and A. Doucet, “COIN: Compression with implicit neural representations,” 2021, *arXiv:2103.03123*.

[29] E. Dupont, H. Loya, M. Alizadeh, A. Golinski, Y. W. Teh, and A. Doucet, “COIN : Neural compression across modalities,” *Trans. Mach. Learn. Res.*, 2022.

[30] H. Jun and A. Nichol, “Shap-e: Generating conditional 3D implicit functions,” 2023, *arXiv:2305.02463*.

[31] Z. Chen and H. Zhang, “Learning implicit fields for generative shape modeling,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5939–5948.

[32] M. Bauer, E. Dupont, A. Brock, D. Rosenbaum, J. R. Schwarz, and H. Kim, “Spatial functa: Scaling functa to imagenet classification and generation,” 2023, *arXiv:2302.03130*.

[33] M. Tancik et al., “Fourier features let networks learn high frequency functions in low dimensional domains,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, vol. 33, pp. 7537–7547 .

[34] J. Zuiderveld, M. Federici, and E. J. Bekkers, “Towards lightweight controllable audio synthesis with conditional implicit neural representations,” in *Proc. NeurIPS 2021 Workshop Deep Generative Models Downstream Appl.*, 2021.

[35] F. Szatkowski, K. J. Piczak, P. Spurek, J. Tabor, and T. Trzciński, “Hypernetworks build implicit neural representations of sounds,” in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, 2023, pp. 661–676.

[36] D. Ha, A. M. Dai, and Q. V. Le, “Hypernetworks,” in *Proc. Int. Conf. Learn. Representations*, 2017.

[37] A. Gulati et al., “Conformer: Convolution-augmented transformer for speech recognition,” in *Proc. Interspeech*, 2020, pp. 5036–5040.

[38] A. Conneau, A. Baevski, R. Collobert, A. Mohamed, and M. Auli, “Unsupervised cross-lingual representation learning for speech recognition,” 2020, *arXiv:2006.13979*.

[39] H. Zen et al., “LibriTTS: A corpus derived from LibriSpeech for text-to-speech,” in *Proc. Interspeech*, 2019, pp. 1526–1530.

[40] Z. Zhang et al., “Speak foreign languages with your own voice: Cross-lingual neural codec language modeling,” 2023, *arXiv:2303.03926*.

[41] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, “Robust speech recognition via large-scale weak supervision,” in *Proc. Int. Conf. Mach. Learn.*, 2023, pp. 28492–28518.