INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Streaming Transformer for Hardware Efficient Voice Trigger Detection and False Trigger Mitigation

Vineet Garg[†*], *Wonil Chang*[†*], *Siddharth Sigtia*[‡], *Saurabh Adya*[†], *Pramod Simha*[†], *Pranay Dighe*[†], *Chandra Dhir*[†]

[†]Apple, USA
[‡]Apple, UK

{vineetgarg, wonil_chang, sidsigtia, sadya, psimha, pdighe, cdhir}@apple.com

Abstract

We present a unified and hardware efficient architecture for two stage voice trigger detection (VTD) and false trigger mitigation (FTM) tasks. Two stage VTD systems of voice assistants can get falsely activated to audio segments acoustically similar to the trigger phrase of interest. FTM systems cancel such activations by using post trigger audio context. Traditional FTM systems rely on automatic speech recognition lattices which are computationally expensive to obtain on device. We propose a streaming transformer (TF) encoder architecture, which progressively processes incoming audio chunks and maintains audio context to perform both VTD and FTM tasks using only acoustic features. The proposed joint model yields an average 18% relative reduction in false reject rate (FRR) for the VTD task at a given false alarm rate. Moreover, our model suppresses 95% of the false triggers with an additional one second of post-trigger audio. Finally, on-device measurements show 32% reduction in runtime memory and 56% reduction in inference time compared to non-streaming version of the model.

Index Terms: Keyword Spotting, Speech Recognition, Acoustic Modeling, Neural Networks, Deep Learning, False Trigger Mitigation

1. Introduction

Intelligent voice assistants (VA) are becoming a ubiquitous part of our daily interaction with smart phones, smart speakers, and other AI-enabled devices. A VA query starts with a trigger phrase followed by the user's request. Many of the on-device VTD systems follow a two stage cascaded design [1][2][3][4] to process the request. The first stage has a low footprint always-on model [5][6][7], which processes continuous audio streams to identify candidate audio segments containing the trigger phrase. Some of these segments can be false triggers that are acoustically similar to the trigger phrase. Since unintended activation of a VA can invade user privacy [8] and degrade user experience, we use a larger capacity model [3][4][9] as a second stage VTD to decide whether to activate the VA or not.

Unlike VTD, FTM systems focus on the post-trigger audio to reduce unintended activation of VAs. It has been shown that post-trigger audio has discriminating information to determine the user intent which can be used to achieve higher accuracy decisions for VTD [10][11][12]. Towards this end, various FTM systems combine acoustic features with automatic speech recognition (ASR) based features [13][14][15][16][17][18] to determine if the user query is intended for the VA or not. Generally, full-fledged ASR systems are available on a server only because they are computationally expensive to be run locally. This raises privacy concerns since unintended queries can leave the user's device. On-device FTM systems use only acoustic features [11][19][20] for hardware efficiency and to avoid such privacy concerns.

Both VTD and FTM tasks aim to reduce the unintended activation of VAs using two different approaches. While VTD has access to the trigger segment, FTM has access to the trigger segment followed by the post-trigger audio. To attain the common goal, we propose a joint streaming TF encoder architecture, which progressively processes incoming audio chunks while maintaining the audio context to perform both VTD and FTM tasks simultaneously. Our proposed model uses only acoustic features for both the tasks and removes dependency on server-side ASR. Morevover, our model shares the compute between VTD and FTM tasks leading to reduced run-time memory and latency. This enables the use of our proposed model on low power edge devices without server-side support. To our knowledge, this work is the first to simultaneously handle the two target tasks on-device in a streaming fashion.

In our experiments, we show that the joint model improves FRR by an average 18% relative compared to the baseline at the same false alarm operating point for the VTD task. The model also correctly rejects up to 95% of unintended false triggers with an additional one second of post-trigger audio for the FTM task. The streaming nature of the model allows reuse of audio context, which reduces the run-time memory by 32% and the inference time by 56% when processing an additional one second of post-trigger audio. Furthermore, the run-time memory and inference time increase linearly with respect to the audio length, whereas they grow quadratically for the non-streaming version of the model.

The rest of the paper is organized as follows. Section 2 provides details of our proposed architecture. Section 3 has the details on training methodology. Section 4 discusses evaluation results and ablation study followed by the conclusions in Section 5.

2. Model Architectures

The outline of the proposed system design with the newly introduced streaming model architecture is depicted in Figure 1. We focus on optimizing the two targets tasks that are interconnected via the backbone TF encoder module (the blue block in Figure 1). The backbone extracts acoustic embeddings, which are used by the phonetic transcription branch for the VTD task and phrase discrimination branch for the FTM task. The third branch is an auto-regressive TF decoder which is used only during training as a regularizer. We first describe details of the

[*] Equal contribution.

http://dx.doi.org/10.21437/Interspeech.2021-1428

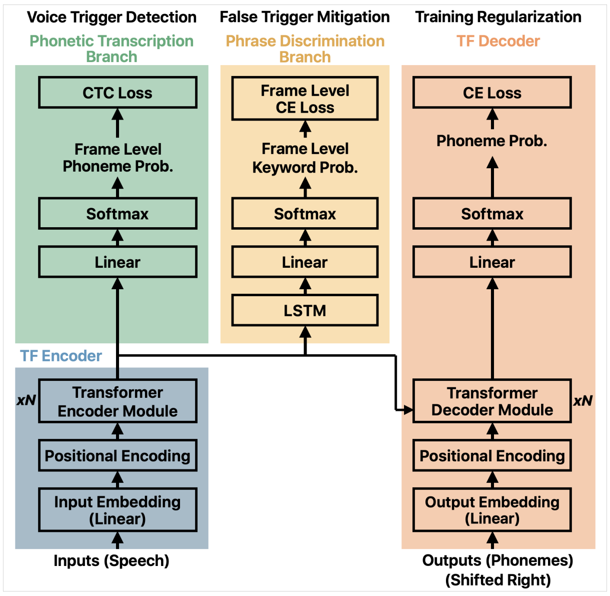


Figure 1: *Proposed network architecture. Blue block is the backbone of TF encoder. Orange block is an optional auto-regressive TF decoder that can be added during training. Green block is the phonetic transcription branch and is trained by minimizing CTC loss. Yellow block is the phrase discrimination branch and is trained by minimizing frame-level CE loss.*

baseline TF encoder architecture [4].

2.1. TF Encoder: Baseline Architecture

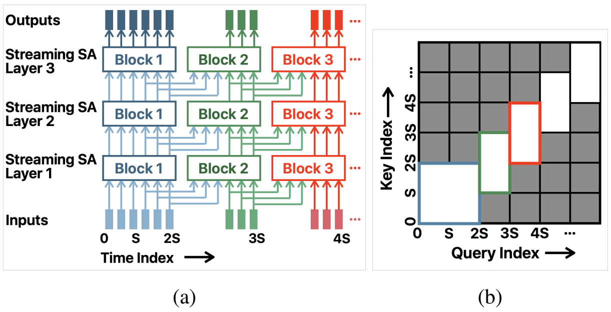
We use the TF encoder [4] as baseline, which is a Transformer [21] model trained with the connectionist temporal classification (CTC) loss on the encoder and cross-entropy (CE) loss on the decoder. We use only the encoder during inference. Since self attention (SA) layers of the encoder look at the entire input context at once, we refer to them as vanilla SA layers in rest of the paper. The trained TF encoder is used to initialize the multi-task learning (MTL) framework, where we add a parallel branch on top of the encoder for phrase discrimination task [3][4][12]. Both the phonetic transcription and phrase discrimination tasks are trained to minimize the CTC loss in their respective branches.

2.2. Streaming TF Encoder: Proposed Architecture

We build upon the existing TF encoder proposed in [4] and introduce a novel streaming TF encoder that: (1) replaces vanilla SA layers with streaming SA layers, (2) introduces frame-wise CE loss in the phrase discrimination branch rather than the CTC loss, and (3) introduces a unidirectional long-short-term memory (uniLSTM) layer in the phrase discrimination branch.

2.2.1. Streaming SA Layers

To enable training and inference of the proposed streaming TF encoder, we replace the vanilla SA layers with streaming SA layers. We follow the block processing protocol presented in [22]. The streaming SA layers process the incoming audio in a blockwise manner with certain shared left context, as shown in Figure 2a. Let $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots]$, $\mathbf{K} = [\mathbf{k}_1, \mathbf{k}_2, \ldots]$, and $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots]$ represent the query, key, and value matrices for a vanilla SA layer, respectively. A streaming SA layer



(a)

(b)

Figure 2: *(a) Block diagram of streaming SA layers. (b) Attention mask to simulate a streaming SA layer. The white regions propagate attention weights. Blue, green, and red rectangle represents the attention weights for Block 1, 2, and 3, respectively. In both (a) and (b), we assume that the block shift S is 50% of the block size B, i.e., $B = 2S$.*

uses $\mathbf{Q}_i$, $\mathbf{K}_i$, and $\mathbf{V}_i$ for the i^{th} block processing. We assume that the block shift S is 50% of the block size B, i.e., $B = 2S$. In the first block, $\mathbf{Q}_1 = [\mathbf{q}_1, \ldots, \mathbf{q}_{2S}]$, $\mathbf{K}_1 = [\mathbf{k}_1, \ldots, \mathbf{k}_{2S}]$, and $\mathbf{V}_1 = [\mathbf{v}_1, \ldots, \mathbf{v}_{2S}]$. For $i \geq 2$, $\mathbf{Q}_i = \left[\mathbf{q}_{iS+1}, \ldots, \mathbf{q}_{(i+1)S}\right]$, $\mathbf{K}_i = \left[\mathbf{k}_{(i-1)S+1}, \ldots, \mathbf{k}_{(i+1)S}\right]$, and $\mathbf{V}_i = \left[\mathbf{v}_{(i-1)S+1}, \ldots, \mathbf{v}_{(i+1)S}\right]$. Note that we use last S vectors of query, last B vectors of key, and last B vectors of value vectors from the second block processing. While $\mathbf{Q}_i$ uses the query vectors from the latest block shift, $\mathbf{K}_i$ and $\mathbf{V}_i$ use the key and value vectors from the previous block shift as well.

We simulate the streaming block processing in a single pass while training by assigning an attention mask (Figure 2b) to attention weight matrix of the vanilla SA layer as in [23]. The mask propagates only the white regions of attention weight matrix and generates the equivalent attention output of a streaming SA described above. This helps avoid slowdown of the training by iterative block processing. The decoder still has vanilla SA and cross attention layers since the decoder is not used during inference. While many of the streaming SA models use relative position encoding [22][24][25], we use an absolute position encoding scheme from the original TF [21].

2.2.2. CE Loss in Phrase Discrimination Branch

In the baseline architecture, we used the sequence to label CTC loss for phrase discrimination [3][4][12]. The CTC loss is prone to making decisions in a greedy manner by producing highly peaky and overconfident distributions [26]. This can cause streaming SA layers trained with CTC loss (in the phrase discrimination branch) to misclassify the false trigger utterances which are acoustically similar to the trigger phrase as true triggers. Post-trigger acoustic context, which can help with accurate phrase discrimination, cannot be used since there is no future audio look-ahead mechanism. This presents an inherent limitation for streaming inference of the phrase discrimination task when the CTC loss is used for training. Therefore, we replace the CTC loss in the phrase discrimination branch with the CE loss. Since the target would have a single label (true or false trigger), we replicate the target labels across all frames and compute frame-wise CE loss.

2.2.3. UniLSTM in Phrase Discrimination Branch

We introduce a uniLSTM [27] layer between the TF encoder embeddings and the dense layer in the phrase discrimination branch. UniLSTM has two benefits here; it smoothens the fluc-

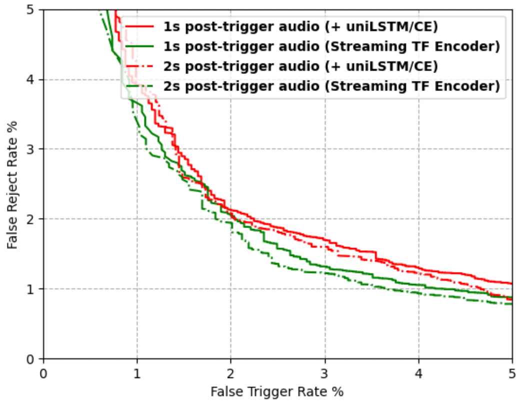


Figure 3: *False Trigger Mitigation with respect to post-trigger audio context.*

tuation of the phrase discrimination output by accumulating the activity of TF encoder embeddings, and it enables long-range context propagation across distant streaming blocks.

3. Training Methodology

The training methodology is similar to our baseline TF encoder [3][4][12]. In the first phase, we train a Transformer network where the encoder has streaming SA layers in a sequence-to-sequence fashion for a phonetic transcription task with the CTC loss on the encoder and CE loss on the decoder (blue, green and orange blocks in Figure 1). The training data has 2700 hours of clean audio with transcribed phonetic labels (54 classes). The audio data is augmented with room impulse responses and echo residuals making a total of 8100 hours of data. We compute 40 dimensional mel-filterbank features at a rate of 100 frames per second. At every frame, we stack the current frame with three frames each for previous and future to create 280 dimensional features. The sequence is sub-sampled by a factor of three and fed to the network. The output of the network is 54 dimensional corresponding to 54 context independent (CI) phones, including start of sequence, end of sequence, word boundary and a blank label. The network has six SA layers with 256 hidden units each and 4 heads with each block followed by 1024 hidden units of a feed forward block. In streaming SA layers, we use a block size $B = 64$ (1.92s) frames and move in steps of $S = 32$ frames with an overlap of previous $B - S = 32$ frames.

In the second phase of training, we remove the TF decoder (orange block in Figure 1) and add a parallel phrase discrimination branch (yellow block in Figure 1) on top of the encoder. The new branch has a uniLSTM with hidden size of 256 followed by a two dimensional linear layer. We jointly train the model for phonetic transcription and phrase discrimination task in MTL manner. While the phonetic training data remains the same as previous phase, we add 40,000 false trigger utterances and 140,000 true trigger utterances for the phrase discrimination task similar to [3][4][11][12].

4. Evaluations and Ablation Study

4.1. Evaluation Data

Our proposed joint model is evaluated on 2 tasks: VTD and FTM. For VTD, we use the same evaluation datasets as used in our baseline system [4]. The first dataset is a structured evaluation set consisting of utterances from 100 participants recorded through a smart speaker with an equal proportion of male and female participants. Each of the utterance begins with the trigger phrase and is obtained under different noise and playback conditions. The negative data consists of 2,000 hours of general speech data which does not have the trigger phrase. The second unstructured dataset is recorded at home by 80 households. Each family uses a smart speaker daily for two weeks. A first-pass DNN-HMM system [28] with a low threshold detects audio segments which are acoustically similar to the trigger phrase, including false triggers. With this dataset, we can measure FRR and false-trigger rates for realistic in-home scenarios similar to in the wild usage. More detailed descriptions on the two evaluation datasets can be found in [3][4]. For the FTM task, we use 88,000 true triggers and 2,800 unintended false triggers as evaluation data.

4.2. Results

4.2.1. Voice Trigger Detection

We evaluate the phonetic transcription branch of four different variants of our model to better understand the impact of each change from the baseline TF encoder to the proposed streaming TF encoder.

- TF encoder
- + Streaming SA: vanilla SA layers in the encoder are replaced with streaming SA layers (Section 2.2.1)
- + uniLSTM / CE: uniLSTM layer is added in the phrase discrimination branch (Section 2.2.3) and the CTC is loss replaced by frame-wise CE loss (Section 2.2.2)
- Streaming TF encoder

Figure 4 shows the Detection Error Trade-off (DET) curves on the structured evaluation set with log(X)-axis representing false alarms (FA) per hour of active audio and Y-axis representing FRR. While changing the vanilla SA layers to streaming SA layers degrades FRR, the introduction of a uniLSTM and CE loss in the phrase discrimination branch improves VTD task accuracy. In fact, our proposed model performs at par with the the case when SA layers are not streaming.

This is an interesting result since structure of the phonetic transcription branch remains unchanged; uniLSTM and CE loss changes were introduced to the phrase discrimination branch. However, since these two branches share a common backbone (blue block in Figure 1), better training of the phrase discrimination due to uniLSTM/CE loss ensures VTD improvements for phonetic transcription branch. Table 1 shows FRR of all four variants at an operating point of 1 FA in 100 hours of active audio.

Figure 5 shows the DET curves for the unstructured evaluation set. We observe a similar trend as seen for the structured evaluation set. Table 1 shows the FRR of all four variants at an operating point of 200 false alarms. We have reported results for a different operating point here compared to [4] since our unstructured evaluation set has increased in past year.

4.2.2. False Trigger Mitigation

Next, we present the results for the FTM task. The phrase discrimination branch of our proposed joint system acts on the post-trigger audio to mitigate the remaining false triggers. The decision score at a given frame is determined by averaging per-frame positive class scores of the previous 10 frames. Figure

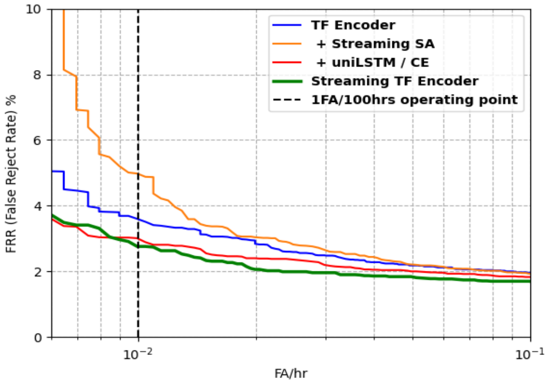


Figure 4: *DET curves for structured evaluation set*

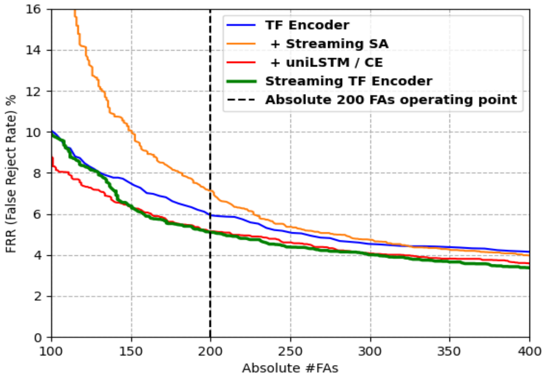


Figure 5: *DET curves for take-home evaluation set*

Table 1: *Voice Trigger Detection performance for 4 models on 2 datasets*

Architecture	Streaming SA Layer	uniLSTM in phrase discrimination	Phrase discrimination loss	FRR (%) on structured dataset[1]	FRR (%) on unstructured dataset[2]
TF Encoder			CTC	3.6	5.9
+ Streaming SA	✓		CTC	4.9	7.1
+ uniLSTM / CE		✓	CE	2.9	5.1
Streaming TF Encoder	✓	✓	CE	**2.8**	**5.1**

[1] FRR at an operating point of 1 FA/100 hrs for structured evaluation set
[2] FRR at an operating point of 200 FAs on unstructured take home evaluation set

Table 2: *False Trigger Mitigation results with respect to post-trigger audio. False Trigger Rate shown at FRR of 1%*

Architecture	1s	2s
TF Encoder	7.0	5.5
+ Streaming SA	7.7	7.7
+ uniLSTM / CE	5.5	4.7
Streaming TF Encoder	**4.2**	**3.7**

3 shows DET curves with the X-axis as false trigger rate and Y-axis as FRR. We can clearly see that post-trigger audio is significantly helping to reduce unintended false triggers. At an additional FRR of 1%, our proposed model can mitigate up to 95.8% of the unintended false triggers with an additional one second of post-trigger audio. Table 2 shows how the false trigger rate improves as more post-trigger audio is provided. In fact, our proposed streaming TF encoder performs at par with the vanilla SA layers.

4.3. Hardware Efficiency

In this section, we show that the proposed streaming TF encoder network is more efficient than a vanilla SA layer based TF encoder in terms of runtime memory usage and reusing the computations to have constant runtime complexity as input audio duration increases. A typical vanilla SA layer has a computational complexity of $O(T^2D)$ where T is the audio length and D is the hidden dimension of SA layers. The proposed streaming architecture converts the quadratic relation with the audio length into linear relation $O(TDS)$ by introducing a constant block shift S. This is particularly useful when analyzing long segments, where the computation can now be distributed in time instead of having one big compute block. This also ensures constant compute requirements at well defined points and hence compute resources can be reserved and scheduled more intelligently. Latency is another aspect to consider when using this technology for FTM. The streaming nature of our proposed model allows us to take decisions at regular intervals and cancel requests as soon as the score drops below a threshold. We compare our proposed model against the case when SA layers process the full audio utterance at once. We use the same on-device experimental setup of a 2019 smart phone as our baseline [4]. Table 3 shows 56% improvement in inference time when processing an additional one second of post-trigger audio. Network latency further improves by 63% as we increase the post-trigger audio to two seconds. Network runtime memory usage also decreases by 32% with additional one second of post-trigger audio and 47% with two seconds of post-trigger audio.

Table 3: *Relative improvements in network runtime latency and memory usage on 2019 smart phone with respect to the additional post-trigger audio context. Our proposed streaming TF encoder is compared against the baseline TF encoder*

Metric	1s	2s
Network runtime memory	32%	47%
Network runtime latency	56%	63%

5. Conclusions

We have presented a unified and hardware-efficient streaming TF encoder architecture for VTD and FTM. We extended our baseline TF encoder by: (1) replacing vanilla SA layers in the encoder with streaming SA layers, (2) introducing frame-wise CE loss rather than the CTC loss in the phrase discrimination branch, and (3) adding a uniLSTM layer in the phrase discrimination branch. The proposed model improves FRR by an average 18% relative and correctly mitigates up to 95% of unintended false triggers with an additional one second of post-trigger audio. Additionally, our model also shares the compute between VTD and FTM tasks leading to 32% reduction in on-device run time memory and 56% reduction in inference time compared to the baseline.

6. References

[1] A. Gruenstein, R. Alvarez, C. Thornton, and M. Ghodrat, "A Cascade Architecture for Keyword Spotting on Mobile Devices," *arXiv preprint arXiv:1712.03603*, 2017.

[2] M. Wu, S. Panchapagesan, M. Sun, J. Gu, R. Thomas, S. N. P. Vitaladevuni, B. Hoffmeister, and A. Mandal, "Monophone-Based Background Modeling for Two-Stage On-Device Wake Word Detection," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[3] S. Sigtia, P. Clark, R. Haynes, H. Richards, and J. Bridle, "Multi-Task Learning for Voice Trigger Detection," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020.

[4] S. Adya, V. Garg, S. Sigtia, P. Simha, and C. Dhir, "Hybrid Transformer/CTC Networks for Hardware Efficient Voice Triggering," in *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association*, 2020.

[5] A. Shrivastava, A. Kundu, C. Dhir, D. Naik, and O. Tuzel, "Optimize What Matters: Training DNN-HMM Keyword Spotting Model Using End Metric," *ArXiv*, vol. abs/2011.01151, 2020.

[6] T. N. Sainath and C. Parada, "Convolutional Neural Networks for Small-Footprint Keyword Spotting," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[7] S. Sigtia, R. Haynes, H. Richards, E. Marchi, and J. Bridle, "Efficient Voice Trigger Detection for Low Resource Hardware," in *Proc. Interspeech 2018*, 2018.

[8] L. Schönherr, M. Golla, T. Eisenhofer, J. Wiele, D. Kolossa, and T. Holz, "Unacceptable, Where is My Privacy? Exploring Accidental Triggers of Smart Speakers," *ArXiv*, vol. abs/2008.00508, 2020.

[9] R. Kumar, M. Rodehorst, J. Wang, J. Gu, and B. Kulis, "Building a Robust Word-Level Wakeword Verification Network," in *INTERSPEECH*, 2020.

[10] J. Wang, R. Kumar, M. Rodehorst, B. Kulis, and S. N. P. Vitaladevuni, "An Audio-Based Wakeword-Independent Verification System," in *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association*, 2020.

[11] P. Dighe, E. Marchi, S. Vishnubhotla, S. Kajarekar, and D. Naik, "Knowledge Transfer for Efficient On-device False Trigger Mitigation," 2020.

[12] S. Sigtia, J. Bridle, H. Richards, P. Clark, E. Marchi, and V. Garg, "Progressive Voice Trigger Detection: Accuracy vs Latency," *ArXiv*, vol. abs/2010.15446, 2020.

[13] S. H. Mallidi, R. Maas, K. Goehner, A. Rastrow, S. Matsoukas, and B. Hoffmeister, "Device-Directed Utterance Detection," in *Proc. Interspeech 2018*, 2018.

[14] R. Maas, A. Rastrow, C. Ma, G. Lan, K. Goehner, G. Tiwari, S. Joseph, and B. Hoffmeister, "Combining Acoustic Embeddings and Decoding Features for End-of-Utterance Detection in Real-Time Far-Field Speech Recognition Systems," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[15] W. Jeon, L. Liu, and H. Mason, "Voice Trigger Detection from LVCSR Hypothesis Lattices Using Bidirectional Lattice Recurrent Neural Networks," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[16] R. Agarwal, X. Niu, P. Dighe, S. Vishnubhotla, S. Badaskar, and D. Naik, "Complementary Language Model and Parallel Bi-LRNN for False Trigger Mitigation," in *Proc. Interspeech 2020*, 2020.

[17] P. Dighe, S. Adya, N. Li, S. Vishnubhotla, D. Naik, A. Sagar, Y. Ma, S. Pulman, and J. Williams, "Lattice-Based Improvements for Voice Triggering Using Graph Neural Networks," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[18] K. Gillespie, I. C. Konstantakopoulos, X. Guo, V. T. Vasudevan, and A. Sethy, "Improving Device Directedness Classification of Utterances with Semantic Lexical Features," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020.

[19] A. Norouzian, B. Mazoure, D. Connolly, and D. Willett, "Exploring Attention Mechanism for Acoustic-based Classification of Speech Utterances into System-directed and Non-system-directed," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[20] X. Tong, C.-W. Huang, S. H. Mallidi, S. Joseph, S. Pareek, C. Chandak, A. Rastrow, and R. Maas, "Streaming ResLSTM with Causal Mean Aggregation for Device-Directed Utterance Detection," *ArXiv*, vol. abs/2007.09245, 2020.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is All You Need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[22] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, "Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context," in *ACL*, 2019.

[23] E. Tsunoo, Y. Kashiwagi, T. Kumakura, and S. Watanabe, "Transformer ASR with Contextual Block Processing," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019.

[24] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer Transducer: A Streamable Speech Recognition Model with Transformer Encoders and RNN-T Loss," *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[25] Y. Wang, H. Lv, D. Povey, L. Xie, and S. Khudanpur, "Wake Word Detection with Streaming Transformers," *ArXiv*, vol. abs/2102.04488, 2021.

[26] H. Liu, S. Jin, and C. Zhang, "Connectionist Temporal Classification with Maximum Entropy Regularization," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] S. Team, "Hey Siri: An On-Device DNN-Powered Voice Trigger for Apple's Personal Assistant," *Apple Machine Learning Journal*, vol. 1, no. 6, 2017. [Online]. Available: https://machinelearning.apple.com/research/hey-siri