

# EXPLORING SEQUENCE-TO-SEQUENCE TRANSFORMER-TRANSDUCER MODELS FOR KEYWORD SPOTTING

Beltrán Labrador<sup>1\*</sup>, Guanlong Zhao<sup>2\*</sup>, Ignacio López Moreno<sup>2\*</sup>  
Angelo Scorza Scarpati<sup>2</sup>, Liam Fowl<sup>2</sup>, Quan Wang<sup>2</sup>

<sup>1</sup>AUDIAS (Audio, Data Intelligence and Speech), Universidad Autónoma de Madrid, Spain  
<sup>2</sup>Google LLC, USA

beltran.labrador@uam.es, {guanlongzhao, elnota, angelos, lfowl, quanw}@google.com

## ABSTRACT

In this paper, we present a novel approach to adapt a sequence-to-sequence Transformer-Transducer ASR system to the keyword spotting (KWS) task. We achieve this by replacing the keyword in the text transcription with a special token  $\langle kw \rangle$  and training the system to detect the  $\langle kw \rangle$  token in an audio stream. At inference time, we create a decision function inspired by conventional KWS approaches, to make our approach more suitable for the KWS task. Furthermore, we introduce a specific keyword spotting loss by adapting the sequence-discriminative Minimum Bayes-Risk training technique. We find that our approach significantly outperforms ASR based KWS systems. When compared with a conventional keyword spotting system, our proposal has similar performance while bringing the advantages and flexibility of sequence-to-sequence training. Additionally, when combined with the conventional KWS system, our approach can improve the performance at *any* operation point.

**Index Terms**— Keyword spotting, sequence-to-sequence models, transformer transducer, speech recognition

## 1. INTRODUCTION

Keyword spotting (KWS), also referred to as spoken term detection, is the task of detecting specific words or multi-word phrases in speech. KWS has wide applications in speech data mining, audio indexing, and phone call routing [1]. Specifically, with the advent of voice assistants in the past few years, keyword spotting has become a common technique to “wake” the voice assistants as a gateway to engage in further conversations with them (e.g. “Okay Google,” “Hey Siri,” or “Alexa”).

The keyword spotting task is closely related to the automatic speech recognition (ASR) task. The main difference between the two tasks is that KWS only focuses on the detection accuracy of a small set of phrases while ASR tries to identify all spoken words in a recording. In recent years, sequence-to-sequence (seq2seq), end-to-end (E2E) trained ASR models, especially those following the RNN-T [2] paradigm have achieved state-of-the-art results in terms of word error rate, e.g., Transformer-Transducer (T-T) [3] and Conformer-Transducer [4].

\*Equal contribution. Beltrán performed this work as an intern at Google.

The authors thank Jason Pelecanos, Alex Park, Doroteo Torre Toledano, Alicia Lozano Diez, Joaquín González Rodríguez, Andre Perunicic, Pierre-Louis Cedoz, Hyun Jin Park, Ding Zhao, Han Lu, Françoise Beaufays, and Pedro Moreno Mengibar. Beltrán was partially supported by FPI RTI2018-098091-B-I00, MCIU/AEI/10.13039/501100011033/FEDER, UE and PID2021-125943OB-I00, MCIN/AEI/10.13039/501100011033/FEDER, UE from the Spanish Ministerio de Ciencia e Innovación, Agencia y del Fondo Europeo de Desarrollo Regional.

Inspired by the quality gains in E2E ASR models, in this work, we explore using Transformer-Transducer for KWS. A straightforward solution to KWS using this technique is to train a T-T model that outputs both the keyword and other spoken tokens. Then the keyword detection can be done by inspecting the presence of the keyword string in the output ASR results. This approach is sub-optimal since the detection accuracy can be easily skewed by minor ASR errors (e.g., mis-recognizing “Okay Google” as “Okay GOOGL”), especially when detecting multi-word and multi-syllable key phrases, or when using grapheme based ASR models. In addition, simply adopting a general purpose ASR model for KWS would suffer from the data sparsity issue given that generally the key phrases appear much less frequently than other spoken words<sup>1</sup>, leading to low detection accuracy.

To mitigate the aforementioned issues, we propose several novel techniques to adapt the T-T ASR model to better fit the KWS task. First, to reduce the negative impact of ASR errors on the keyword detection performance, we constrain the T-T model to treat the keyword audio segment as a coherent acoustic event. The model outputs a single keyword token  $\langle kw \rangle$  instead of the entire keyword string when detecting a keyword in the input audio stream. To achieve this, we modify the text transcription of the spoken utterance by replacing the keywords with the special token  $\langle kw \rangle$ , and then train the model to output both regular text tokens and the special keyword tokens. Second, we propose a model training loss that directly minimizes the KWS error rates as a solution to the data sparsity issue. Lastly, we formulate a keyword confidence score as the model’s output instead of parsing the ASR decoding results to make the KWS prediction, which allows the T-T based KWS model to perform on different operation points by adjusting its prediction threshold, offering flexibility for different application scenarios.

We conduct an extensive performance comparison between the proposed T-T based KWS model and a suite of strong conventional non-ASR [7] and ASR-based KWS systems with different model configurations. Overall, we find this system provides better KWS performance than the baselines. We also observe that the proposed T-T based KWS model is complimentary to conventional non-ASR KWS models, enabling system fusion for use cases that require higher level of detection accuracy.

The rest of the paper is organized as follows. In Section 2, we compare the proposed work with related prior works. Section 3 describes the proposed modeling strategies. We introduce the baselines systems in Section 4. We then offer the experimental setup and results in Sections 5 and 6, respectively. Finally, we discuss and highlight the key findings of this work in Section 7.

<sup>1</sup>Similar to the rare words issue [5, 6] in E2E ASR.

## 2. RELATION TO PRIOR WORK

The traditional approaches to solving the keyword spotting problem use the Hidden Markov Model (HMM), which is improved by using deep neural networks (DNNs) to characterize the acoustic features [8–12]. In [13], the authors train a DNN to classify each frame into either the keyword or the “filler” audio and then apply a posterior handling method to produce a final confidence score. This method is refined in [14] by adopting CNN networks to take into account both time and frequency relationships of the speech signal, and also in [15–17] by using recurrent neural networks that can help capture longer temporal dependencies in the speech sequence. E2E approaches [7, 18] further improve detection accuracy and lower the resource requirements, by directly producing a keyword spotting score and using a compact neural network topology. The proposed work differs from these conventional approaches by using a seq2seq model to capture longer-term dependencies and avoid the need of using forced-alignment to produce frame-level training labels, since the seq2seq model implicitly learns an alignment between the input acoustic and output text sequences.

Several works have also explored using seq2seq models for detecting keywords in continuous speech. In [19], the authors propose a LSTM encoder followed by a CTC decoder to train a keyword spotting system that creates phoneme lattices for efficient search. In [20], N-best results are indexed for searching using attention models. In [21], an E2E model is trained to detect a keyword without using an explicit speech recognizer. In [22], the authors propose a way to bias a general purpose ASR model towards a specific keyword of interest. In contrast with previous approaches, our proposed model uses a probability score from the softmax output as in [7], avoids the need of N-best results that require a more computationally expensive inference process, and allows us to introduce direct optimization of the keyword token during training (Section 3.2).

## 3. METHOD

### 3.1. TT-KWS: Transformer-Transducer Keyword Spotting

During the last several years, transducers have shown state-of-the-art performance on speech recognition tasks [3]. The T-T model consists of an audio encoder that converts input features into acoustic embedding vectors, a label encoder that converts text tokens into linguistic embedding vectors, and a joint network that takes the acoustic and linguistic embedding vectors as the input and outputs a probability distribution over a set of predefined text tokens. In this work, we use an audio encoder constructed with Transformer blocks [23], and a label encoder comprising LSTM layers. The joint network is composed of fully-connected layers and outputs graphemes.

For the KWS task, we treat the speech segment that corresponds to the keyword as a single coherent acoustic event. Our main proposal, referred by us as TT-KWS, is to edit the ASR labels at training time, substituting every keyword appearance by a special token  $\langle kw \rangle$  that is part of our grapheme vocabulary, inspired by the speaker turn detection ideas in [24] where speaker diarization is improved by using a T-T to detect speaker changes.

At inference time, our TT-KWS system outputs this special token when it detects a keyword in the audio stream. For the keyword spotting task, we can ignore all the other tokens and focus only on the special  $\langle kw \rangle$  token. This approach can be easily adapted to any other keywords with minimal changes to the model training pipeline, adding flexibility to the keyword spotting modeling process. A model trained for one keyword can be used as a pre-trained model for a different keyword, without having to use a force alignment method to create fine-grain labels, saving from possible force alignments errors and also simplifying the possible path for federat-

ed/ephemeral [25] training of KWS models.

### 3.2. MBR training

To further optimize the model for the KWS task during training, inspired by the Minimum Bayes-Risk (MBR) training technique [26–28], we present a training loss that directly optimizes the recognition accuracy of the keyword token. The idea is to first compute the KWS false negative (FN) and false positive (FP) rates on the N-best hypotheses (produced by a beam search) during training, and then formulate a training loss that minimizes the expected FN and FP rates.

Mathematically, let  $\mathbf{H}_{ij}$  be the  $j$ -th hypothesis of the  $i$ -th training sample,  $\mathbf{P}_{ij}$  be the probability score associated with the  $\mathbf{H}_{ij}$  hypothesis, and  $\mathbf{R}_i$  be the reference transcription for all the hypotheses of the  $i$ -th training sample. To adapt MBR training to the keyword spotting task, we count the number of the special keyword token  $\langle kw \rangle$  in  $\mathbf{H}_{ij}$  and  $\mathbf{R}_i$ , referred to as  $\mathbf{K}_{ij}^H$  and  $\mathbf{K}_i^R$ , respectively. We then calculate the number of keyword token insertions ( $\mathbf{FP}_{ij}$ ) and deletions ( $\mathbf{FN}_{ij}$ ) as follows,

$$\mathbf{FP}_{ij} = \max(0, \mathbf{K}_{ij}^H - \mathbf{K}_i^R), \mathbf{FN}_{ij} = \max(0, \mathbf{K}_i^R - \mathbf{K}_{ij}^H), \quad (1)$$

and compute the per sample loss as

$$\mathbf{L}_{ij} = \mathbf{P}_{ij} \cdot \frac{\alpha \mathbf{FP}_{ij} + \beta \mathbf{FN}_{ij}}{\mathbf{K}_i^R + \epsilon}, \quad (2)$$

where  $\alpha$  and  $\beta$  control the relative strength of each sub-component and  $\epsilon$  is a small constant value to avoid numeric errors. We note that for any particular training utterance, only one of  $\mathbf{FP}_{ij}$  and  $\mathbf{FN}_{ij}$  is non-zero, but across the entire training set we expect a more diverse distribution. Finally we can compute the per batch training loss as

$$\mathbf{L} = \sum_i \sum_j \mathbf{L}_{ij} - \lambda \log P(\mathbf{Y}|\mathbf{X}), \quad (3)$$

where  $-\log P(\mathbf{Y}|\mathbf{X})$  is the negative log probability of the reference transcript  $\mathbf{Y}$  conditioned on the input acoustic features  $\mathbf{X}$ . For simplicity, we refer the negative log probability loss as the RNN-T loss. The regularization term  $\lambda$  controls the strength of the RNN-T loss.

### 3.3. Scoring method

KWS systems often output a confidence score such that different thresholds can be applied to make the final detection decision based on application scenarios. Therefore, instead of making a binary decision based on the keyword appearance in the ASR result, we take the softmax output value of the  $\langle kw \rangle$  token at the end of the joint network as the KWS score, which is a direct measure of the network’s confidence on the keyword at a given frame. To obtain the score of the system on an utterance, we take the maximum score that the network outputs for the entire utterance.

## 4. BASELINES

### 4.1. Baseline 1: End-to-end KWS

To compare our proposal to a state-of-the-art system, we train an E2E KWS baseline [7] that uses stacked Singular Value Decomposition Filters (SVDFs) to approximate fully-connected layers with a low rank decomposition. This model is optimized for low-resource use cases. More details of this baseline can be found in [7].

### 4.2. Baseline 2: ASR based KWS

We construct an ASR system for the KWS task as another baseline. This ASR model has the same architecture as the proposed TT-KWS model but is trained with the regular RNN-T loss, predicting the original text transcription verbatim.

*Bigram edit distance scoring:* We compute a KWS score for this baseline to allow a fairer comparison with the other systems. For every bigram in the hypothesis text, we compute its grapheme-level ASR edit distances (ignoring spaces and cases) against the two keywords of interest (“Hey Google” and “Okay Google”). We then find the bigram with the minimum edit distance  $\text{GED}_{\min}$  and use  $e^{-\text{GED}_{\min}}$  as the KWS score. For example, for hypothesis “Okay GOOGL,” its KWS score is  $e^{-1} \approx 0.37$  since it contains one deletion error. This score provides tolerance towards miss-spelled keyword recognition.

## 5. EXPERIMENTAL SETUP

In all experiments and for all systems, we use the same train and evaluation datasets, feature front-end, and data augmentation. When handling user data, we abide by Google AI principles [29] and Privacy principles [30].

### 5.1. Evaluation metrics

To compare the systems on different operating points, we use Detection Error Trade-off (DET) curves, where different False Negative (FN) rate vs. False Positive (FP) rate operating points are displayed by varying the detection threshold. We also report the Equal Error Rate (EER), where the FP and FN are equal. In addition, we present the results at specific FP points of interest, where the FP rate is relatively low (0.5% and 1%). The reason is that in applications we are more willing to trade higher FN rates for a lower FP rate but not vice versa. A high FP rate would trigger the keyword too often and hinder the user experience.

### 5.2. Data description and preparation

Our dataset is composed of vendor collected speech data that contains different English accents (from US, India, UK and Australia), varying acoustic conditions (e.g. inside vehicles, phone speech, and background noises), balanced gender distribution, and equally distributed near-field and far-field audio. The data is divided into positive and negative utterances. A positive utterance contains any of the keywords “Hey Google” or “Okay Google” followed by a speech query. A negative utterance does not contain any of the keywords. Negative utterances include around 15k of the so-called confusable utterances, with words that are phonetically close to the target keywords. We split the dataset for training, validation and evaluation purposes. The training set has 4300h of positive data (4M utterances) and 4000h of negative data (3.6M utterances). The training set is augmented by artificially corrupting clean utterances using a room simulator, adding varying degrees of noise and reverberation [31], producing 25 additional augmented versions for each utterance. The validation set, which is used for training monitoring and checkpoint selection, contains 700h of positive and 500h of negative data. The test set contains 3800h of positive and 7000h of negative data, also including confusable utterances.

For each utterance, we extract 40-dim log-Mel filter-bank energies from a 25ms window, stack every 4 frames, and sub-sample every 3 frames, to produce a 160-dim feature vector with a stride of 30 ms as the input to the audio encoder.

### 5.3. System configurations

#### 5.3.1. End-to-end KWS baseline

We train two configurations for the baseline E2E KWS system [7]. The *Baseline KWS small* configuration has 330K parameters with seven SVDF layers, where the first four SVDF layers act as an encoder and the last three SVDF layers act as a decoder. Each of the

**Table 1:** Hyper-parameters of a Transformer block.

	Large	Small
Input feature projection	160	160
Dense layer 1	1024	128
Dense layer 2	256	32
Number attention heads	8	8
Head dimension	64	64
Dropout ratio	0.1	0.1

encoder SVDF layers has 576 nodes with a memory of 6 and is followed by a 64-dim projection layer except for the last encoder layer, which is followed by a 32-dim projection layer. Each of the decoder SVDF layers has 32 nodes with a memory of 24 and is followed by a 32-dim projection layer except for the last decoder layer, which is followed by a final projection layer to predict a binary KWS label. On top of the *Baseline KWS small* configuration, the *Baseline KWS large* configuration increases the encoder SVDF layers to have 4096 nodes and a 128-dim projection layer except that the last encoder SVDF layer has a 16-dim projection layer. The decoder SVDF layers remain unchanged, resulting in 3.9M parameters.

#### 5.3.2. Transformer-Transducer based configurations

For both the ASR based KWS baseline and the proposed TT-KWS model, we compare two size configurations. The audio encoder consists of Transformer blocks, see Table 1 for the hyper-parameters of each block under different configurations.

Specifically, for the *large* configurations, referred in Table 2 as *ASR baseline large*, *TT-KWS large* and *TT-KWS + MBR large*, the audio encoder consists of 15 Transformer blocks, and the label encoder is a 128-dim LSTM layer. The joint network projects both the audio and label encoder outputs to two embedding vectors of the same size (i.e., the size of the audio encoder’s output); the two embedding vectors are then summed and projected to a probability distribution over the 75 output graphemes (the English alphabet, punctuation, the keyword token  $\langle k_w \rangle$ , and special characters such as “\$”). This configuration has around 13M parameters.

On the other hand, the *small* configurations referred in Table 2 as *ASR baseline small*, *TT-KWS small* and *TT-KWS + MBR small*, have seven smaller Transformer blocks with hyper-parameters detailed in Table 1. We reduce their label encoders to a 32-dim LSTM layer, resulting in a total number of 350k trainable parameters.

To train the models with the RNN-T loss we use the same hyper-parameters as in [3]. Afterwards, to further optimize to the KWS task with the proposed MBR loss, we warm-start the *TT-KWS + MBR* models with the *TT-KWS* models trained with the RNN-T loss, and then fine-tune them with a combination of the RNN-T and MBR losses as described in Eq. (3) following a fixed learning rate of  $10^{-5}$ . Empirically, we set the hyper-parameters  $\{\alpha, \beta, \lambda\}$  in Eqs. (2) and (3) to  $\{1.0, 1.0, 0.01\}$  and compute the MBR loss on the 4-best hypotheses, produced by a beam search of size 8.

## 6. RESULTS AND DISCUSSION

We summarize the evaluation results in terms of EER as well as other key operating points of interest (1% FP and 0.5% FP) in Table 2.

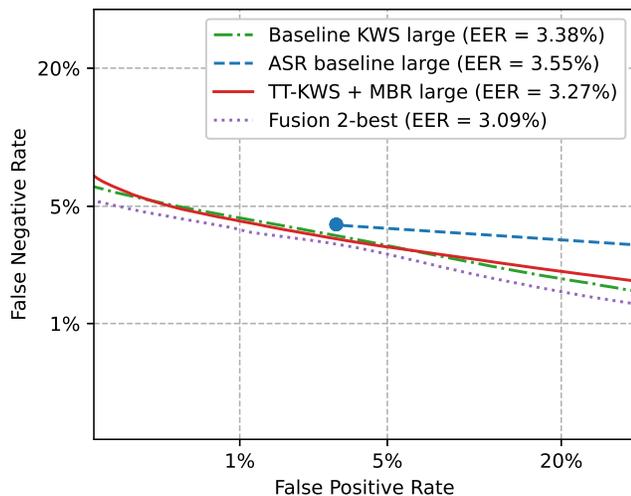
For the low-resource condition (i.e., the “small” models), the baseline E2E KWS model trained with the SVDF layers achieves the best performance. Among the T-T based KWS systems, we observe a 31% relative EER improvement (8.24% vs. 5.68%) when modifying the T-T model to output the special keyword token  $\langle k_w \rangle$  instead of the entire keyword string (e.g., “Okay Google”), which shows that

**Table 2:** Performance of the systems under different operation points. For the ASR baselines, they do not have operation points that can achieve a 1% or 0.5% false positive rate. The “Fusion 2-best” configuration combines the “Baseline KWS large” and “TT-KWS + MBR large” models.

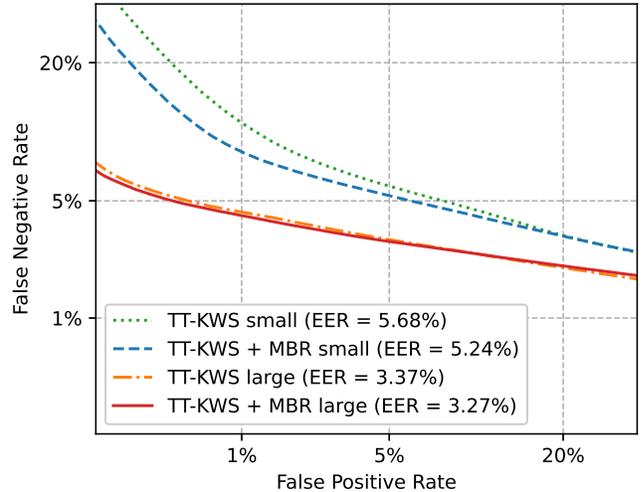
	EER	FN @ 1% FP	FN @ 0.5% FP
Baseline KWS small	<b>3.78%</b>	<b>5.07%</b>	<b>5.78%</b>
ASR baseline small	8.24%	-	-
TT-KWS small	5.68%	11.68%	17.49%
TT-KWS + MBR small	5.24%	8.70%	12.18%
Baseline KWS large	3.38%	4.34%	4.91%
ASR baseline large	3.55%	-	-
TT-KWS large	3.37%	4.35%	5.01%
TT-KWS + MBR large	<b>3.27%</b>	<b>4.17%</b>	<b>4.78%</b>
Fusion 2-best	<b>3.09%</b>	<b>3.75%</b>	<b>4.25%</b>

by treating the keyword audio segment as a coherent acoustic event we can constrain the model to focus on predicting the keyword holistically. Adding the MBR training loss further improve the EER by 7.7% relative (5.68% vs. 5.24%). More importantly, within the low FP region, the MBR training loss is more effective, reducing the FN rate by 25.5% (1% FP) and 30.4% (0.5% FP) relatively, compared with the models trained with only the RNN-T loss. The MBR loss minimizes the expected FN and FP rates during training, and the results here show that this effect can be translated to unseen data.

For the condition that allows more modeling resources (i.e., the “large” models), the proposed TT-KWS model trained with the MBR loss achieves the best performance overall. In Fig. 1, we observe that the *TT-KWS + MBR large* and *Baseline KWS large* systems perform similarly on all operation points, with the proposed *TT-KWS + MBR large* model achieving a moderate 3.3% relative EER improvement. Both systems outperform the *ASR baseline large* configuration by a large margin on every operation point. We note that the *bigram edit distance scoring* method described in Section 4.2 enables us to compute the DET curve of the *ASR baseline large* system, extending it from operating on a single FN/FP configuration (i.e., performing an exact match between the predicted keyword string and the reference transcript.) to a more flexible score-based approach.



**Fig. 1:** Detection error trade-off (DET) curves of various KWS systems. The dot on the DET curve of the *ASR baseline large* model shows its performance without the *bigram edit distance scoring* method (Section 4.2).



**Fig. 2:** Detection error trade-off curves of the TT-KWS models trained with the MBR loss.

We are also interested in the effectiveness of the MBR training loss in different conditions. As shown in Fig. 2, in general, MBR is able to improve system performance at every operation point. The benefit of this modeling technique is more significant when the number of parameters is limited. We note that this technique is especially effective in the low FP operation region. This result suggest that the MBR training technique can concentrate the modeling capacity on the main task of KWS when the resource is more limited.

Finally, we perform a system fusion by summing up the KWS scores of two individual systems *Baseline KWS large* and *TT-KWS + MBR large* and then make the prediction. The results are shown in the last row of Table 2. On the EER score, we observe a significant 8.6% (vs. *Baseline KWS large*) and 5.5% (vs. *TT-KWS + MBR large*) relative performance gain compared with the individual systems. Within the low FP regions, the improvements are even larger, resulting in up to 13.6% relative FN rate improvement (4.34% vs. 3.75% at 1% FP) compared with the individual models. In addition, based on Fig. 1, the system fusion consistently outperforms the individual systems on all operation points. These results suggest that the proposed TT-KWS model is complementary to a conventional E2E KWS system.

## 7. CONCLUSION

This paper shows that a TT-KWS model can achieve results that are comparable with or better than conventional KWS models, especially when resources are less constrained. Additionally, our results suggest that for CPU and battery constrained devices such as lower capability phones, the conventional KWS models remain the best option. KWS models that utilize ASR techniques often require large model sizes and computationally intensive decoding process, which make it challenging to deploy on mobile devices that have small battery and restricted computational power. In these use cases, we prefer small footprint models to be able to listen and process audio continuously. On the other hand, power plugged devices (e.g., smart speakers, smart displays, and vehicles), where computational power is less restricted, enable us to use more complex T-T based KWS systems. These systems allow us to take advantage of larger datasets and increase the diversity of the keyword phrases. A system combination between the conventional KWS and TT-KWS yields the best performance, making the system fusion solution suitable for applications that require more accurate results.

## 8. REFERENCES

- [1] Ivan Lopez Espejo, Zheng-Hua Tan, John Hansen, and Jesper Jensen, “Deep spoken keyword spotting: An overview,” *IEEE Access*, vol. 10, pp. 4169–4199, Jan. 2022.
- [2] Alex Graves, “Sequence transduction with recurrent neural networks,” *arXiv preprint arXiv:1211.3711*, 2012.
- [3] Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar, “Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss,” in *Proc. ICASSP*, 2020, pp. 7829–7833.
- [4] Anmol Gulati et al., “Conformer: Convolution-augmented transformer for speech recognition,” in *Proc. Interspeech*, 2020, pp. 5036–5040.
- [5] Florian Lux and Ngoc Thang Vu, “Meta-learning for improving rare word recognition in end-to-end ASR,” in *Proc. ICASSP*, 2021, pp. 5974–5978.
- [6] Chao-Han Huck Yang, Linda Liu, Ankur Gandhe, Yile Gu, Anirudh Raju, Denis Filimonov, and Ivan Bulyko, “Multi-task language modeling for improving speech recognition of rare words,” in *Proc. ASRU*, 2021, pp. 1087–1093.
- [7] Raziel Alvarez and Hyun-Jin Park, “End-to-end streaming keyword spotting,” in *Proc. ICASSP*, 2019, pp. 6336–6340.
- [8] Sankaran Panchapagesan, Ming Sun, Aparna Khare, Spyros Matsoukas, Arindam Mandal, Björn Hoffmeister, and Shiv Vitaladevuni, “Multi-task learning and weighted cross-entropy for DNN-based keyword spotting,” in *Proc. Interspeech*, 2016, pp. 760–764.
- [9] Ming Sun et al., “Compressed time delay neural network for small-footprint keyword spotting,” in *Proc. Interspeech*, 2017, pp. 3607–3611.
- [10] Kenichi Kumatani, Sankaran Panchapagesan, Minhua Wu, Minjae Kim, Nikko Strom, Gautam Tiwari, and Arindam Mandal, “Direct modeling of raw audio with DNNs for wake word detection,” in *Proc. ASRU*, 2017, pp. 252–257.
- [11] Jinxi Guo, Kenichi Kumatani, Ming Sun, Minhua Wu, Anirudh Raju, Nikko Ström, and Arindam Mandal, “Time-delayed bottleneck highway networks using a DFT feature for keyword spotting,” in *Proc. ICASSP*, 2018, pp. 5489–5493.
- [12] Minhua Wu, Sankaran Panchapagesan, Ming Sun, Jiacheng Gu, Ryan Thomas, Shiv Naga Prasad Vitaladevuni, Björn Hoffmeister, and Arindam Mandal, “Monophone-based background modeling for two-stage on-device wake word detection,” in *Proc. ICASSP*, 2018, pp. 5494–5498.
- [13] Guoguo Chen, Carolina Parada, and Georg Heigold, “Small-footprint keyword spotting using deep neural networks,” in *Proc. ICASSP*, 2014, pp. 4087–4091.
- [14] Tara N. Sainath and Carolina Parada, “Convolutional neural networks for small-footprint keyword spotting,” in *Proc. Interspeech*, 2015, pp. 1478–1482.
- [15] Ming Sun et al., “Max-pooling loss training of long short-term memory networks for small-footprint keyword spotting,” in *Proc. SLT*, 2016, pp. 474–480.
- [16] Sercan Ö. Arik, Markus Kliegl, Rewon Child, Joel Hestness, Andrew Gibiansky, Chris Fougner, Ryan Prenger, and Adam Coates, “Convolutional recurrent neural networks for small-footprint keyword spotting,” in *Proc. Interspeech*, 2017, pp. 1606–1610.
- [17] Santiago Fernández, Alex Graves, and Jürgen Schmidhuber, “An application of recurrent neural networks to discriminative keyword spotting,” in *International Conference on Artificial Neural Networks*. Springer, 2007, pp. 220–229.
- [18] Hyun-Jin Park, Patrick Violette, and Niranjan Subrahmanya, “Learning to detect keyword parts and whole by smoothed max pooling,” in *Proc. ICASSP*, 2020, pp. 7899–7903.
- [19] Yimeng Zhuang, Xuankai Chang, Yanmin Qian, and Kai Yu, “Unrestricted vocabulary keyword spotting using LSTM-CTC,” in *Proc. Interspeech*, 2016, pp. 938–942.
- [20] Andrew Rosenberg, Kartik Audhkhasi, Abhinav Sethy, Bhuvana Ramabhadran, and Michael Picheny, “End-to-end speech recognition and keyword search on low-resource languages,” in *Proc. ICASSP*, 2017, pp. 5280–5284.
- [21] Kartik Audhkhasi, Andrew Rosenberg, Abhinav Sethy, Bhuvana Ramabhadran, and Brian Kingsbury, “End-to-end ASR-free keyword search from speech,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 11, no. 8, pp. 1351–1359, 2017.
- [22] Yanzhang He, Rohit Prabhavalkar, Kanishka Rao, Wei Li, Anton Bakhtin, and Ian McGraw, “Streaming small-footprint keyword spotting using sequence-to-sequence models,” in *Proc. ASRU*, 2017, pp. 474–481.
- [23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, vol. 30.
- [24] Wei Xia, Han Lu, Quan Wang, Anshuman Tripathi, Yiling Huang, Ignacio Lopez Moreno, and Hasim Sak, “Turn-to-diarize: Online speaker diarization constrained by transformer transducer speaker turn detection,” in *Proc. ICASSP*, 2022, pp. 8077–8081.
- [25] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtarik, Ananda Theertha Suresh, and Dave Bacon, “Federated learning: Strategies for improving communication efficiency,” in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.
- [26] Karel Veselý, Arnab Ghoshal, Lukáš Burget, and Daniel Povey, “Sequence-discriminative training of deep neural networks,” in *Proc. Interspeech*, 2013, pp. 2345–2349.
- [27] Rohit Prabhavalkar et al., “Minimum word error rate training for attention-based sequence-to-sequence models,” in *Proc. ICASSP*, 2018, pp. 4839–4843.
- [28] Chao Weng, Chengzhu Yu, Jia Cui, Chunlei Zhang, and Dong Yu, “Minimum bayes risk training of RNN-Transducer for end-to-end speech recognition,” in *Proc. Interspeech*, 2020, pp. 966–970.
- [29] Google, “Artificial intelligence at Google: Our principles,” <https://ai.google/principles>, Accessed: 2022-10-17.
- [30] “Google’s privacy principles,” <https://googleblog.blogspot.com/2010/01/googles-privacy-principles.html>, Accessed: 2022-10-17.
- [31] Chanwoo Kim et al., “Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in google home,” in *Proc. Interspeech*, 2017, pp. 379–383.