INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

PhonMatchNet: Phoneme-Guided Zero-Shot Keyword Spotting for User-Defined Keywords

Yong-Hyeok Lee, Namhyun Cho

Speech AI Lab., NCSOFT Corporation, South Korea
{eug92, cnh2769}@ncsoft.com

Abstract

This study presents a novel zero-shot user-defined keyword spotting model that utilizes the audio-phoneme relationship of the keyword to improve performance. Unlike the previous approach that estimates at utterance level, we use both utterance and phoneme level information. Our proposed method comprises a two-stream speech encoder architecture, self-attention-based pattern extractor, and phoneme-level detection loss for high performance in various pronunciation environments. Based on experimental results, our proposed model outperforms the baseline model and achieves competitive performance compared with full-shot keyword spotting models. Our proposed model significantly improves the EER and AUC across all datasets, including familiar words, proper nouns, and indistinguishable pronunciations, with an average relative improvement of 67% and 80%, respectively. The implementation code of our proposed model is available at https://github.com/ncsoft/PhonMatchNet.

Index Terms: keyword spotting, user-defined, zero-shot, open-vocabulary

1. Introduction

Keyword spotting (KWS) in speech processing enables the identification of specific keywords in audio signals. Generally, conventional KWS methods require a significant amount of labeled data for training, which is a limitation in scenarios in which target keywords are rare, specialized, or user-defined [1]. Recently, as user-friendly services, such as smart speakers, AI assistants, and personalized digital humans have become widespread, the demand for personalized technology that allows consumers to define their keywords instead of using predetermined keywords set by manufacturers has increased.

Zero-shot KWS is key to addressing this challenge, enabling a model to detect user-defined keywords with no prior training on those keywords. Most user-defined KWS (UDKWS) models are implemented using the query-by-example (QbyE) approach, which compares an input speech and a pre-registered speech in latent space [2, 3, 4]. Despite the success of the QbyE approach, it is limited in terms of performance and implementation because it compares the input speech with the pre-enrolled speech.

Recently, a cross-modal correspondence detector (CMCD) that compares speech and text and can omit speech registration when adding new keywords has been proposed [5]. However, this method does not properly distinguish pairs of similar pronunciations, as the similarity between speech and text is calculated at the utterance level.

In this paper, we propose a novel zero-shot UDKWS model to address these issues. Our proposed model includes a two-stream speech encoder with a pre-trained speech embedder [6], self-attention-based pattern extractor [7], and phoneme-level detection loss to achieve high performance in various pronunciation environments. We conducted experiments on datasets containing familiar words [8], proper nouns [9], and indistinguishable pronunciations [5] to evaluate the effectiveness of our proposed model. Our proposed model outperforms the baseline model and achieves competitive performance compared with full-shot keyword spotting models. Particularly, our proposed model achieved a significant improvement in the equal error rate (EER) and area under the curve (AUC) across all datasets, with an average relative improvement of 67% and 80%, respectively.

2. Related works

This section examines recent UDKWS methods and related studies for our proposed method. The most recent model [5] showed degraded performance in similar pronunciations that are difficult to distinguish owing to the comparison of speech-text pairs at the utterance level. Furthermore, the audio encoder cannot properly handle uncommon pronunciations in the training dataset because it comprises only fully trainable modules. Thus, we employed a pre-trained speech embedder and phoneme-level detection loss to overcome these limitations.

Recently, various studies on non-semantic representation that allow a single embedder model to be used across various target tasks, not only in speech recognition [10] but also in other areas, such as keyword spotting [6], speaker identification, and language identification [11] have been presented. These studies improve models using techniques, such as self-supervised or unsupervised learning, multi-task learning [12], and knowledge distillation [13]. Our objective is to obtain embeddings that can show high performance in general speech situations; therefore, we selected and froze the embedder model [6] while considering the KWS performance and model size.

In various detection tasks using speech-text [14], visual-text [15], speech-visual [16], and visual-only [17], temporal location information of targets significantly assists in achieving high performance. However, providing the correct labels for every timestamp in sequence data is challenging. Therefore, connectionist temporal classification (CTC) loss [18], which does not require explicit alignment information, is widely used. However, unlike speech recognition in which speech and text labels always match, CTC loss is not appropriate for UDKWS, where speech and text labels may not match. Therefore, we propose a method that uses the sameness of speech and text pronunciation as a label rather than the exact time information.

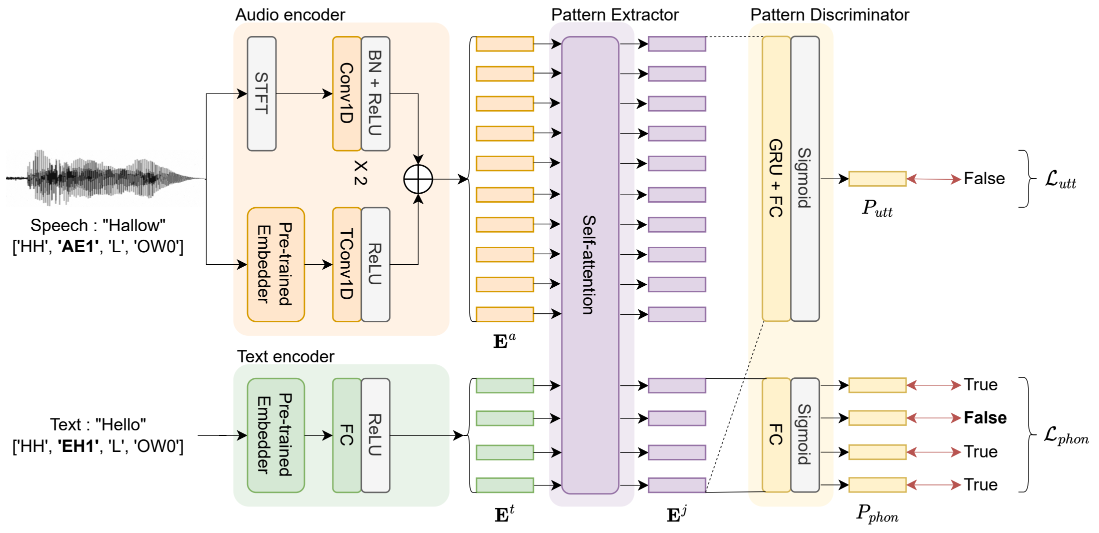

Figure 1: *Architecture of the proposed model. Boldface and red arrows denote the unmatched phonemes, labels, and BCE losses. "TConv" is the abbreviation of "Transposed Convolution."*

3. Proposed method

Here, we describe our proposed model architecture and training criterion, as shown in Figure 1. Our proposed model comprises three sub-modules: an audio and text encoder with a pre-trained embedder, pattern extractor, and pattern discriminator. We use two loss functions based on the cross-entropy loss in the training criterion, which are computed at the utterance ($\mathcal{L}_{utt}$) and phoneme levels ($\mathcal{L}_{phon}$).

3.1. Model architecture

Audio Encoder. The audio encoder comprises two feature extractors: a pre-trained speech embedder [6] with high performance in representing general pronunciations and a fully trainable feature extractor, which learns representations of special pronunciations, such as proper nouns. The pre-trained speech embedder has 775 ms windows and computes 96-dimensional feature vectors every 80 ms. We upsample the feature and time dimensions using a 1-D transposed convolution with a kernel size of 5 and stride of 4, along with fully connected layers. The fully trainable feature extractor comprises two 1-D convolutions with a kernel size of 3, batch normalization, and ReLU layers. The first convolution layer has a stride of 2, whereas the others have a stride of 1. The 40-dimensional mel-filterbank coefficients used as input are extracted every 10 ms with a window of 25 ms. Finally, we compute the 128-dimensional feature vectors extracted every 20 ms by adding the two feature vectors. We denote audio embeddings as $\mathbf{E}^a \in \mathbb{R}^{T_a \times 128}$, where T_a and 128 are the lengths of the audio and embedding dimension, respectively.

Text Encoder. The text encoder, similar to [5], includes a pre-trained grapheme-to-phoneme (G2P) model[1] followed by a fully connected layer and a ReLU activation function. We extract the G2P embedding from the last hidden states of the encoder. We denote text embeddings by $\mathbf{E}^t \in \mathbb{R}^{T_t \times 128}$, which is the same as the audio encoder.

Pattern extractor. Considering the characteristics of the KWS task, which requires maintaining fewer model parameters, our pattern extractor is based on a self-attention [7, 15] rather than a cross-attention mechanism [16, 19]. As in [20], during the fusion of multiple modalities, the self-attention method does not require other modules, unlike other attention mechanisms. The matrix of attention outputs for a set of queries Q with keys and values packed into matrices K and V is computed as

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{1}$$

Unimodal embeddings $\mathbf{E}^a$ and $\mathbf{E}^t$ are concatenated along the time dimension to prepare the joint embeddings $\mathbf{E}^j$:

$$\mathbf{E}^c = (\mathbf{E}^a; \mathbf{E}^t) \in \mathbb{R}^{(T_a+T_t)\times 128}. \tag{2}$$

The concatenated embedding $\mathbf{E}^c$ is calculated as joint embeddings $\mathbf{E}^j$ using equation 1:

$$\mathbf{E}^j = \text{Attention}(\mathbf{E}^c, \mathbf{E}^c, \mathbf{E}^c) \in \mathbb{R}^{(T_a+T_t)\times 128}. \tag{3}$$

We used a lower triangular matrix as an attention mask to use only causal information of the intra-modality.

Pattern discriminator. Our pattern discriminator determines two probabilities, the match probability of audio and keywords and the audio and phoneme. To detect an utterance-level matching, we use a single GRU layer with 128 dimensions, taking the joint embeddings $\mathbf{E}^j$ along the time dimension as an input. The output of the last hidden state is fed into a fully connected layer with a sigmoid function:

$$P_{utt} = \sigma(\mathbf{W}^u \cdot \text{GRU}(\mathbf{E}^j) + \mathbf{b}^u) \in \mathbb{R}^{1\times 1}. \tag{4}$$

Similarly, we extract only the phoneme sequences from the sequence of joint embeddings $\mathbf{E}^j$ and feed this into the fully connected layer with a sigmoid function to detect phoneme-level matching:

$$P_{phon} = \sigma(\mathbf{W}^p \cdot \mathbf{E}^j_{\mathbf{s}} + \mathbf{b}^p) \in \mathbb{R}^{T_t\times 1}, \tag{5}$$

where $\mathbf{W}$, $\mathbf{b}$, σ, and $\mathbf{s}$ denote the trainable weights and biases of each fully connected layer, the sigmoid function, and the frame index of $(T_a, T_a + T_t]$, respectively.

[1]https://github.com/Kyubyong/g2p

Table 1: *Performance of the baseline, proposed method, and ablations on various datasets. † refers to our implementation of H.-K. Shin et al.* $\mathbf{G}$*,* $\mathbf{Q}$*,* $\mathbf{LP_E}$*, and* $\mathbf{LP_H}$ *refer to our test sets from Section 4.1.* ***w/o Phoneme-level Detection Loss*** *represents the case of training using only utterance-level detection loss.* ***w/o Self-attention-based Pattern Extractor*** *means the pattern extractor is composed of cross-modality attention rather than self-attention.* ***w/o Fully trainable Speech Encoder*** *means the audio encoder comprises only a pre-trained speech embedder without convolution layers. We adjust the model parameters of the ablation studies to be the same as the proposed model.*

Method	Params	EER (%) ↓				AUC (%) ↑			
		$\mathbf{G}$	$\mathbf{Q}$	$\mathbf{LP_E}$	$\mathbf{LP_H}$	$\mathbf{G}$	$\mathbf{Q}$	$\mathbf{LP_E}$	$\mathbf{LP_H}$
† CMCD [5]	653K	31.94	26.09	10.48	29.34	73.86	82.42	95.63	77.60
Proposed	655K	**6.77**	**4.75**	**2.80**	**18.82**	**98.11**	**98.90**	99.29	**88.52**
w/o Phoneme-level Detection Loss		7.52	6.05	3.39	19.65	97.45	98.45	**99.30**	88.04
w/o Self-attention-based Pattern Extractor	655K	11.32	18.24	3.47	20.51	95.73	90.14	99.26	87.49
w/o Fully trainable Speech Encoder		14.34	22.90	5.13	25.77	93.91	84.19	98.71	82.53

3.2. Training criterion

Our training criterion ($\mathcal{L}_{total}$) comprises two binary-cross entropy (BCE) losses: an utterance- ($\mathcal{L}_{utt}$) and a phoneme-level detection losses ($\mathcal{L}_{phon}$),

$$\mathcal{L}_{total} = \mathcal{L}_{utt} + \mathcal{L}_{phon}. \tag{6}$$

Utterance-level detection loss. Similar to general KWS methods [21], the utterance-level detection loss is used to train the similarity between speech and keywords within a sample. The sample-level ground truth is used with P_{utt} from Equation 4 to calculate the BCE loss.

Phoneme-level detection loss. We propose the phoneme-level detection loss to improve the performance of distinguishing similar pronunciations (e.g., "friend" and "trend") without using speech and pronunciation alignment information. The phoneme-level ground truth $\mathbf{y}$ is defined as 1 if the phoneme sequence of the speech label is the same as that of the keyword label and 0, otherwise.

$$y_p = \begin{cases} 1 & \text{if } y_p^t = y_p^s \\ 0 & \text{otherwise} \end{cases}, \tag{7}$$

where y_p^s, y_p^t, and p denote the speech, keyword label, and phoneme index from 1 to T_t. As with $\mathcal{L}_{utt}$, the phoneme-level ground truth $\mathbf{y}$ and prediction P_{phon} are used to calculate the BCE loss.

4. Experiments

Here, we describe the experimental setup, including the datasets, evaluation metrics, and implementation details for training and testing. Although most of our experimental settings are similar to those in [5], we employ different methods to create anchor-negative pairs in certain test sets. An example of this is listed in Table 2 for clarity.

4.1. Datasets and metrics

We used three datasets: LibriPhrase [5], Google Speech Commands V1 ($\mathbf{G}$) [8], and Qualcomm Keyword Speech dataset ($\mathbf{Q}$) [9], for training and evaluation. In the training phase, we used the training set of LibriPhrase and babble noise from the MS-SNSD dataset [22] for robust detection. Detailed training conditions were similar to that of [5]. During evaluation, we used the datasets $\mathbf{G}$, $\mathbf{Q}$, LibriPhrase-easy ($\mathbf{LP_E}$), and LibriPhrase-hard ($\mathbf{LP_H}$). $\mathbf{LP_E}$ and $\mathbf{LP_H}$ are datasets reclassified as easy and difficult to differentiate between anchor and negative pairs in the test sets of LibriPhrase [5]. We evaluated our proposed models by measuring the EER and AUC at the sample level on these datasets. Because the datasets $\mathbf{G}$ and $\mathbf{Q}$ do not provide negative pairs, we calculated the EER and AUC by considering all keywords except the positive pairs as negative pairs among the candidate keywords for each dataset. Table 2 provides examples of anchor and negatives in $\mathbf{G}$ and $\mathbf{Q}$.

Table 2: *Examples of anchor and negatives of each dataset*

Dataset	Anchor	Negatives
G	go	yes no up down left right on off stop
Q	hi galaxy	hey android hey snapdragon hi lumina

4.2. Implementation details

Our implementation was based on the TensorFlow library. The training criterion was optimized using the Adam optimizer with the default parameters. The models were trained for 100 epochs with a fixed learning rate of 10^{-3}, and the best model was selected based on performance on the test sets. For training, we used a single V100 with a batch size of 2048 for a day.

5. Results

5.1. Comparison with baseline

We re-implemented the baseline model, CMCD, to compare its performance with that of our proposed model. According to Table 1, our reconstructed CMCD performed similarly in $\mathbf{LP_E}$ and $\mathbf{LP_H}$ but showed decreased performance in $\mathbf{G}$ and $\mathbf{Q}$ datasets compared with the results of the original study [5]. This can be attributed to the difference in the method of generating anchor-negative pairs. Although the method differs from [5], our method can measure the performance of these models with increased accuracy because it computes one anchor and other keywords. Our proposed model outperformed the baseline by a significant margin in all datasets, including familiar words (*e.g.*, "yes" and "no"), proper nouns (*e.g.*, "snapdragon" and "lumina"), and indistinguishable pronunciations (*e.g.*, "friend" and "frind"). A relative improvement of 67% in EER and 80% in AUC was observed on average across all the datasets.

5.2. Ablation Study

We evaluated the effectiveness of our proposed methods as shown in Table 1: phoneme-level detection loss, self-attention-based modality fusion, and two-stream speech encoder.

Effectiveness of Speech Encoder Components. Our speech encoder comprised fully trainable and pre-trained modules. Comparing the first and last rows of Table 1, we observed a considerable improvement in EER and AUC when using only the pre-trained speech embedder, except for the $\mathbf{Q}$ dataset. This could be attributed to the limitation that the embedder model we used was trained on general conversations from YouTube [6]. Therefore, we compensated for this by adding fully trainable modules, which improved the performance on all test sets, including the $\mathbf{Q}$ dataset, as shown in the fourth row.

Fusion Strategy for Audio-Text Representations. Comparing the third and fourth rows of Table 1, the performance of cross-modality attention and that of self-attention-based pattern extractor differed. Unlike the aforementioned analysis, we observed performance improvements on all test sets, particularly on the $\mathbf{Q}$ dataset. The keywords of $\mathbf{Q}$ comprise proper nouns that are uncommon in typical training datasets. Unlike cross-modality attention-based models, a self-attention-based pattern extractor concatenated bi-modal data to produce outputs that contain all information in speech-to-speech, text-to-speech, and text-to-text modalities. This permitted the attention outputs, which are enhanced with speech information, to help distinguish the pronunciation similarity of rare cases in the training dataset.

Effectiveness of the Phoneme-level Detection Loss. We proposed the phoneme-level detection loss to improve discrimination performance in speech-keyword pairs with duplicate pronunciations. As shown in Table 1 and Figure 2, our proposed loss function outperformed the utterance-level detection loss only, particularly on the $\mathbf{LP_H}$ dataset and with low normalized Levenshtein distance, indicating the effectiveness of our proposed loss in discriminating speech-keyword pairs with similar pronunciations.

5.3. Performance analysis via keyword similarity

As shown in Figure 2, we examined the relationship between phonetic similarity and model performance. We defined the similarity between speech-text label pairs using the normalized Levenshtein distance [23] and calculated the mean squared error (MSE) between our model predictions and the labels over the similarity range greater than 0 and less than 1. Comparing our proposed model to the baseline, we observed superior performances across all distances. Notably, the error of the baseline model remained constant for normalized Levenshtein distances below 0.45, whereas our proposed model exhibited a linear improvement. Furthermore, we can attribute the performance improvements in similar pronunciations to the effectiveness of our phoneme-level detection loss.

5.4. Comparison with Conventional Methods

We compared the performance of our proposed zero-shot KWS model with conventional full-shot KWS models in Table 3. Because our model only calculated the matching probability, we

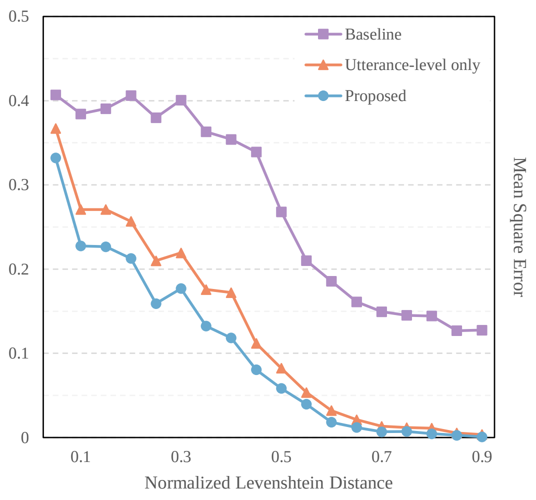

Figure 2: *Evaluation results according to the normalized Levenshtein distance in a LibriPhrase test set. The smaller the distance, the more similar the pronunciation.*

Table 3: *Comparison of conventional KWS models and the proposed model in the test set of Google Speech Commands V1-12.*

Model	Zero-shot	Accuracy ↑
DS-CNN [24]		95.4
Att-RNN [25]		95.6
TC-ResNet [26]		96.6
MHAtt-RNN [27]	✗	97.2
MatchBoxNet [28]		97.5
KWT-3 [29]		97.5
BC-ResNet-8 [30]		**98.0**
Proposed	✓	**96.8**

set a threshold of 0.8 to compute the accuracy. Our proposed model outperformed some conventional models and achieved competitive performance compared with the state-of-the-art model. This result is noteworthy considering that we did not include the Google Speech Command dataset when training our model.

6. Conclusions

This study presented a novel user-defined keyword spotting model that leveraged the relationship between audio and phonemes of keywords. We introduced a two-stream speech encoder containing pre-trained embedders, a self-attention-based pattern extractor, and a phoneme-level detection loss to improve the performance of distinguishing similar pronunciations. Our proposed model outperformed the baseline model and achieved competitive performance compared with full-shot KWS models. However, despite our various efforts, relative performance improvements in challenging cases, such as $\mathbf{LP_H}$ were lower compared with other test sets. Improving the performance of highly similar pronunciations is our future research direction.

7. References

[1] J. Huang, W. Gharbieh, Q. Wan, H. S. Shim, and H. C. Lee, "QbyE-MLPMixer: Query-by-Example Open-Vocabulary Keyword Spotting using MLPMixer," in *Proc. Interspeech 2022*, 2022, pp. 5200–5204.

[2] G. Chen, C. Parada, and T. N. Sainath, "Query-by-example keyword spotting using long short-term memory networks," in *Proc. ICASSP 2015*, 2015, pp. 5236–5240.

[3] L. Lugosch, S. Myer, and V. S. Tomar, "Donut: Ctc-based query-by-example keyword spotting," *NeurIPS 2018 - Workshop on Interpretability and Robustness in Audio, Speech, and Language (IRASL)*, 2018.

[4] J. Huang, W. Gharbieh, H. S. Shim, and E. Kim, "Query-by-example keyword spotting system using multi-head attention and soft-triple loss," in *Proc. ICASSP 2021*, 2021, pp. 6858–6862.

[5] H.-K. Shin, H. Han, D. Kim, S.-W. Chung, and H.-G. Kang, "Learning Audio-Text Agreement for Open-vocabulary Keyword Spotting," in *Proc. Interspeech 2022*, 2022, pp. 1871–1875.

[6] J. Lin, K. Kilgour, D. Roblek, and M. Sharifi, "Training keyword spotters with limited and synthesized speech data," in *Proc. ICASSP 2020*, 2020, pp. 7474–7478.

[7] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, "What does bert with vision look at?" in *ACL (short)*, 2020.

[8] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[9] B. Kim, M. Lee, J. Lee, Y. Kim, and K. Hwang, "Query-by-example on-device keyword spotting," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2019, pp. 532–538.

[10] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.

[11] J. Shor, A. Jansen, R. Maor, O. Lang, O. Tuval, F. de Chaumont Quitry, M. Tagliasacchi, I. Shavitt, D. Emanuel, and Y. Haviv, "Towards learning a universal non-semantic representation of speech," in *Proc. Interspeech 2020*, 2020, pp. 140–144.

[12] J. Peplinski, J. Shor, S. Joglekar, J. Garrison, and S. N. Patel, "FRILL: A non-semantic speech embedding for mobile devices," in *Proc. Interspeech 2021*, 2021, pp. 1204–1208.

[13] J. Shor and S. Venugopalan, "Trillsson: Distilled universal paralinguistic speech representations," in *Proc. Interspeech 2022*, 2022, pp. 356–360.

[14] B. Yusuf, A. Gok, B. Gundogdu, and M. Saraclar, "End-to-end open vocabulary keyword search," in *Proc. Interspeech 2021*, 2021.

[15] K. R. Prajwal, L. Momeni, T. Afouras, and A. Zisserman, "Visual keyword spotting with attention," in *32nd British Machine Vision Conference 2021, BMVC 2021, Online*, 2021, p. 380.

[16] L. Momeni, T. Afouras, T. Stafylakis, S. Albanie, and A. Zisserman, "Seeing wake words: Audio-visual keyword spotting," in *31st British Machine Vision Conference 2020, BMVC 2020, Online*, 2020.

[17] K. R. Prajwal, H. Bull, L. Momeni, S. Albanie, G. Varol, and A. Zisserman, "Weakly-supervised fingerspelling recognition in british sign language videos," in *British Machine Vision Conference*, 2022.

[18] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[19] J. Lee, S.-W. Chung, S. Kim, H.-G. Kang, and K. Sohn, "Looking into your speech: Learning cross-modal affinity for audio-visual speech separation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1336–1345.

[20] H. Chefer, S. Gur, and L. Wolf, "Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 397–406.

[21] I. López-Espejo, Z.-H. Tan, J. H. Hansen, and J. Jensen, "Deep spoken keyword spotting: An overview," *IEEE Access*, vol. 10, pp. 4169–4199, 2021.

[22] C. K. Reddy, E. Beyrami, J. Pool, R. Cutler, S. Srinivasan, and J. Gehrke, "A scalable noisy speech dataset and online subjective test framework," *Proc. Interspeech 2019*, pp. 1816–1820, 2019.

[23] V. I. Levenshtein *et al.*, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, no. 8. Soviet Union, 1966, pp. 707–710.

[24] S. Mittermaier, L. Kürzinger, B. Waschneck, and G. Rigoll, "Small-footprint keyword spotting on raw audio data with sinc-convolutions," in *Proc. ICASSP 2020*, 2020, pp. 7454–7458.

[25] D. Seo, H.-S. Oh, and Y. Jung, "Wav2kws: Transfer learning from speech representations for keyword spotting," *IEEE Access*, vol. 9, pp. 80 682–80 691, 2021.

[26] S. Choi, S. Seo, B. Shin, H. Byun, M. Kersner, B. Kim, D. Kim, and S. Ha, "Temporal convolution for real-time keyword spotting on mobile devices," in *Proc. Interspeech 2019*, 2019, pp. 3372–3376.

[27] O. Rybakov, N. Kononenko, N. Subrahmanya, M. Visontai, and S. Laurenzo, "Streaming keyword spotting on mobile devices," in *Proc. Interspeech 2020*, 2020, pp. 2277–2281.

[28] S. Majumdar and B. Ginsburg, "Matchboxnet: 1d time-channel separable convolutional neural network architecture for speech commands recognition," in *Proc. Interspeech 2020*, 2020, pp. 3356–3360.

[29] A. Berg, M. O'Connor, and M. T. Cruz, "Keyword transformer: A self-attention model for keyword spotting," in *Proc. Interspeech 2021*, 2021, pp. 4249–4253.

[30] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," in *Proc. Interspeech 2021*, 2021, pp. 4538–4542.