Few-Shot Keyword Spotting in Any Language

Mark Mazumder[1], Colby Banbury[1], Josh Meyer[§2], Pete Warden[3], Vijay Janapa Reddi[1]

[1]Harvard University
[2]Coqui
[3]Google

{markmazumder,cbanbury}@g.harvard.edu, josh@coqui.ai, petewarden@google.com,
vj@eecs.harvard.edu

Abstract

We introduce a few-shot transfer learning method for keyword spotting in any language. Leveraging open speech corpora in nine languages, we automate the extraction of a large multilingual keyword bank and use it to train an embedding model. With just five training examples, we fine-tune the embedding model for keyword spotting and achieve an average F_1 score of 0.75 on keyword classification for 180 new keywords unseen by the embedding model in these nine languages. This embedding model also generalizes to new languages. We achieve an average F_1 score of 0.65 on 5-shot models for 260 keywords sampled across 13 new languages unseen by the embedding model. We investigate streaming accuracy for our 5-shot models in two contexts: keyword spotting and keyword search. Across 440 keywords in 22 languages, we achieve an average streaming keyword spotting accuracy of 85.2% with a false acceptance rate of 1.2%, and observe promising initial results on keyword search.

Index Terms: speech recognition, keyword spotting, low-resource languages

1. Introduction

Training keyword spotting (KWS) models requires the manual collection and curation of thousands of target samples across a diverse pool of speakers and accents for each keyword of interest [1] — a prohibitive requirement for under-resourced languages. In this paper, we relax the training data for a KWS model to just five training examples in any language.

We train an embedding model on keyword classification using Common Voice's [2] multilingual crowd-sourced speech dataset, by applying forced alignment [3] to automatically extract 760 frequent words across nine languages. We then fine-tune this embedding model to classify a target keyword with just five sample utterances, even if the model has never seen the target language before. We evaluate our embedding representation's performance on 440 keywords across 22 languages to demonstrate the generalization of our approach to languages and words previously unseen by the embedding model.

Our contributions are as follows: (1) we show promising 5-shot keyword detection accuracy across 22 languages via a multilingual embedding representation; (2) we find that multilingual embeddings improve accuracy and generalize to new languages; (3) we demonstrate that general speech recognition datasets can be repurposed for KWS through forced alignment, highlighting the value of crowd-sourced data in low-resource
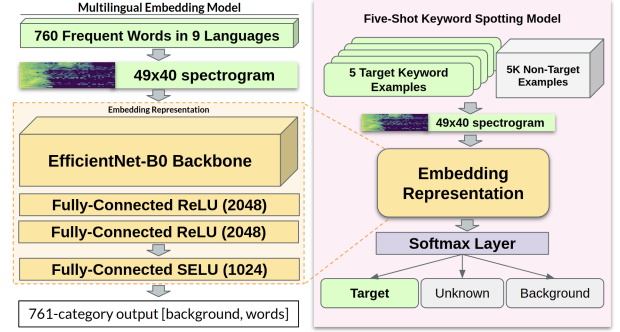
(a) *Multilingual embedding model*   (b) *5-shot keyword spotting*

Figure 1: ***Multilingual Embedding Representation:*** **(a)** *To learn a multilingual embedding for keyword feature extraction, we train a classifier on 760 keywords totaling 1.4M samples in nine languages, and use the output of the penultimate layer of our classifier as a feature vector for arbitrary keywords in any language.* **(b)** *To train a new KWS model, we fine-tune a 3-category classifier using just 5 target examples and 128 non-target samples from a precomputed "unknown" keyword bank.*

settings; and (4) we open-source our code and models and provide a Colab for easy reproduciblity and extension.[1]

These contributions lay the groundwork towards a fully automated, rapid time-to-solution pipeline for generating voice-based command interfaces for arbitrary keywords in low-resource languages. Our current aim is to enable a volunteer to record just 5 examples of a target keyword in a zero-resource language and obtain a robust multi-speaker KWS model. Our existing approach can run on mobile devices, and our longer-term efforts will target deployment on low-cost, memory-constrained, power-efficient microcontrollers to enable always-on KWS support.

2. Related Work

Many approaches to keyword spotting have been proposed. Prior art focused on developing small footprint models for keyword spotting tasks using deep neural networks [4], convolutional neural networks [5, 6, 7, 8], and long short-term memory neural networks [9]. Similarly, we use a CNN based architecture for our classifier, however, these previous methods require thousands of samples of the target keyword. In contrast, our work only requires 5 keyword samples in a target language, which enables keyword spotting in low resource languages.

---

[1]https://github.com/harvard-edge/multilingual_kws

For KWS in low-resource languages, existing methods [10, 11] employ multilingual bottleneck features, dynamic time warping, and autoencoders, requiring roughly 30 samples per keyword in addition to a few hours of untranscribed data. In [12] the authors demonstrate high KWS accuracy with 3 examples for English and Korean. Bluche et al. [13] shows promising accuracy in English for a zero-shot approach to detecting any keyword, and San et al. [14] explores spoken term search across ten languages. Our work utilizes a simpler embedding model and considers few-shot performance across a comparatively large number of languages and speakers, including languages in which our training data only consists of 5 samples of a target keyword. A recent approach to data generation for KWS utilizes speech synthesis [15] for training an embedding model. In comparison, our work targets languages which lack sufficient data for synthesis.

3. Multilingual Keyword Spotting

In this section, we describe our system for training a multilingual embedding model, performing transfer learning, and automating the extraction of a large keyword dataset.

3.1. Multilingual Embedding Model

Our network architecture is summarized in Fig. 1a. We repurpose the output of the penultimate layer of a simple keyword classifier as our embedding representation in our few-shot experiments. Our classifier uses TensorFlow Lite Micro's [16] microfrontend spectrogram [17] as 49x40x1 inputs. It contains approximately 11 million parameters and consists of a randomly-initialized EfficientNet-B0 implementation from Keras [18], followed by a global average pooling layer, two dense layers of 2048 units with ReLU activations, and a penultimate 1024-unit SELU activation [19] layer, before the classifier's 761-category softmax output. We chose SELU activations for their self-normalizing properties. The classifier is trained on 760 words across nine languages (listed in Table 1) and 1.4 million samples in total (Sec. 3.3). We select the most common words in each language and filter by character length of 3 or higher to discard brief words and stop words. Each extraction is padded with silence to one second in length. We also include a background noise category in the output, with 10% of all training samples consisting solely of noise sampled from background noise examples in Google's Speech Commands dataset [1]. Keyword samples are augmented with random 100ms timeshifts, background noise multiplexed at 10% SNR, and SpecAugment [20].

3.2. Few-shot Transfer Learning

For 5-shot transfer learning (Fig. 1b), we use five target samples to fine-tune a 3-class softmax layer (with *target, unknown,* and *background* categories) on the output feature vector of the embedding layers, along with 128 non-target samples drawn from a precomputed bank of 5,000 "unknown" utterances in the nine embedding languages. When training KWS models in languages not seen by the embedding model (e.g., in Welsh), non-target samples are still drawn from this bank, i.e., to train a KWS model in Welsh a user would only need to collect 5 target samples of a Welsh keyword, without also needing to collect non-target examples in Welsh. The weights in the embedding layers are frozen when fine-tuning; we only update the softmax layer. Across 256 total training samples, approximately 45% are in the target category (random augmentations of the five target examples using the same strategy as Sec. 3.1), 45% are

negative samples drawn from the precomputed set of non-target words, and 10% are background noise (Sec. 5.1.2).

3.3. Dataset Generation

Our extracted keywords are entirely sourced from Common Voice [2]. We use the Montreal Forced Aligner [3] to estimate word-level alignments for each < audio,transcript > pair in Common Voice.[2] We performed forced alignment from a flat start only on the data itself, with no prior acoustic models or external data. We automate keyword extraction using the alignment timings. For our experiments, we extracted 4,383,489 samples across 3,126 keywords in 22 languages.

4. Experiments

We evaluate our capabilities for KWS in multiple languages through classification and streaming accuracy experiments.

4.1. Classification Accuracy

We assess classification performance for the embedding model, and for five-shot KWS models evaluated on a large number of extracted target and non-target keywords.

4.1.1. Multilingual embeddings trained on extracted keywords

In order to evaluate the quality of our embedding representation trained on extracted keywords (Sec 3.1), we assess the top-one accuracy of the classifier on a validation set of 161,700 samples. Furthermore, we report the validation accuracy for each language to inspect whether it is skewed. We train the embedding model for 94 epochs using the Adam optimizer from Keras with a learning rate of 0.001.

We also inspect the potential domain gap between extracted and manually recorded keywords. We cross-compare the test accuracy of a *tinyconv* model [21] trained on the keyword "left" chosen randomly from the Google Speech Commands (GSC) dataset [1] and a model trained on keyword extractions of "left" from Common Voice English data. After training a *tinyconv* model on Common Voice data, we assess the model's classification performance on GSC data and vice versa.

4.1.2. Monolingual vs. multilingual embeddings

We explore the accuracy of a multilingual embedding relative to individual language embedddings, by comparing KWS models fine-tuned on each. We train six monolingual embedding models by selecting 165 frequent words per language and using a penultimate layer width of 192 units. We evaluate KWS models for 20 *out-of-vocabulary* words (i.e., words unseen when training the embedding representation) in the target language. We select up to 2,000 samples per keyword and train a KWS model by fine-tuning on 5 samples. We evaluate classification performance by validating on the remaining 1,995 samples as positive class examples, along with 30,000 samples across 90 non-target words as negative examples. To assess keyword classification accuracy for the multilingual embedding, for each of the nine languages in the embedding, we randomly select 20 target words distinct from the 760 keywords used to train the multilingual representation (Sec. 3.1), and evaluate classification performance for 5-shot models against all other positive samples of each keyword and 30,000 negative samples across 90 non-target words. Negative examples are divided evenly be-

---

[2]https://github.com/JRMeyer/common-voice-forced-alignments

tween keywords previously used to train the embedding model (these should now be categorized as non-targets by the 5-shot model, and not misclassified as the target), keywords sampled from the bank of unknown samples used when finetuning 5-shot models (Sec. 3.2), and prevously unencountered keywords which are novel to both the embedding and KWS models.

4.1.3. Out-of-embedding classification

We investigate whether the multilingual embedding can be used to perform keyword spotting in languages not seen by the embedding model, i.e., we establish whether a precomputed multilingual embedding can generalize to other languages without collecting additional training data in that language beyond five samples of a target keyword. We consider classification accuracy across two settings: (1) *out-of-vocabulary* words not previously seen by the multilingual embedding model, but spoken in the languages used to train the embedding, and (2) *out-of-embedding* words in languages unseen when training the multilingual embedding model.

4.2. Few-Shot Streaming Accuracy

In practice, KWS models operate on a continuous stream of audio, thus we inspect streaming accuracy in two regimes. (1) We concatenate individual words with an average gap of 2 seconds (filled with random background noise), to simulate wake-word or command interaction with a voice assistant. For each KWS model, we evaluate on 10 minutes of audio containing approximately 100 keywords and 100 random non-target words. (2) We search for keywords in continuous spoken audio, by concatenating approximately 20 minutes of full sentences from Common Voice. In each case we consider out-of-vocabulary and out-of-embedding words chosen at random across 22 languages. Our streaming post-processing approach follows Sec. 7.2 in [1].

5. Results

We summarize our classification and streaming accuracy evaluations for 5-shot KWS models. We note that in all of our automated KWS evaluations, the five training samples are randomly selected from forced alignment extractions and are not manually inspected beforehand, hence performance for some models will be negatively affected by poor extractions or errors in the original Common Voice recordings.

5.1. Classification Accuracy Results

We evaluate (1) the training performance of our multilingual embedding model, (2) improvements to KWS classification accuracy when using a multilingual embedding representation versus monolingual embeddings, and (3) KWS classification accuracy in languages outside of the multilingual embedding.

5.1.1. Embedding model accuracy

Table 1 summarizes the accuracy of our multilingual embedding model (Sec. 3.1) on a validation set for each of the nine languages used in training. The embedding model achieves an overall classification accuracy of 79.81%.

As a microbenchmark to compare extracted samples versus manually recorded samples, Table 2 reports our cross comparison of two *tinyconv* models trained on Common Voice extractions and GSC data for the keyword "left." The model trained on Common Voice extractions performs worse on the GSC test set than vice versa, dropping to roughly 78%. Future work will

Table 1: ***Classification Accuracy for Multilingual Embedding Model***. *We show the number of words per language and number of training samples the embedding model was trained on, followed by the number of validation samples and the validation accuracy of the embedding model for each language.*

Language	# words	# train	# val	val acc
English	265	518760	57640	78.95
German	152	287100	31900	79.90
French	105	205920	22880	79.16
Kinyarwanda	68	134640	14960	73.64
Catalan	80	132660	14740	87.63
Persian	35	69300	7700	85.70
Spanish	31	61380	6820	79.65
Italian	17	31680	3520	81.16
Dutch	7	13860	1540	72.60
Model	760	1455300	161700	79.81

Table 2: ***Domain gap*** *between our extracted keyword dataset (Extracted) and the manually recorded Google Speech Commands (GSC). The row indicates the training dataset for the model; the column indicates the testing dataset.*

Training ╲ Test	GSC	Extracted
GSC	93.42%	90.49%
Extracted	78.07%	92.23%

seek to reduce this apparent domain gap.

5.1.2. Monolingual vs multilingual embedding results

We find that using the multilingual embedding in Sec. 5.1.1 improves KWS classification accuracy versus monolingual embeddings. Fig. 2 shows receiver operating characteristic (ROC) curves where the false positive rate is visualized against the true positive rate as the threshold for the target keyword is varied for each 5-shot KWS model (Fig. 1b). Twenty previously unseen words were chosen as random KWS targets for each language evaluated. Each model was fine-tuned on 256 samples across 4 epochs with a batch size of 64 (Sec 3.2). The number of samples, epochs, batches, and batch size were chosen empirically via a hyperparameter sweep.

Fig. 2a shows classification accuracy for 5-shot models using six monolingual embedding models. Fig. 2b shows the classification accuracy achieved on 5-shot models using the multilingual embedding representation. Accuracy improves when using the multilingual embedding representation. With an empirically chosen threshold of 0.8, the average (unweighted) F_1 score across all KWS models increases from 0.58 to 0.75. For example, despite there being no additional data in Kinyarwanda between Fig. 2a and Fig. 2b, classification accuracy for Kinyarwanda still improves. This suggests that keyword classification accuracy improves for each language through generalizable features from other languages, as was suggested in [10, 22].

5.1.3. Out-of-embedding classification results

Fig. 2c depicts classification accuracy for 20 common words chosen randomly from each of 13 languages unobserved when training the embedding model. Accuracy remains high for the majority of these languages, with an average F_1 score of 0.65
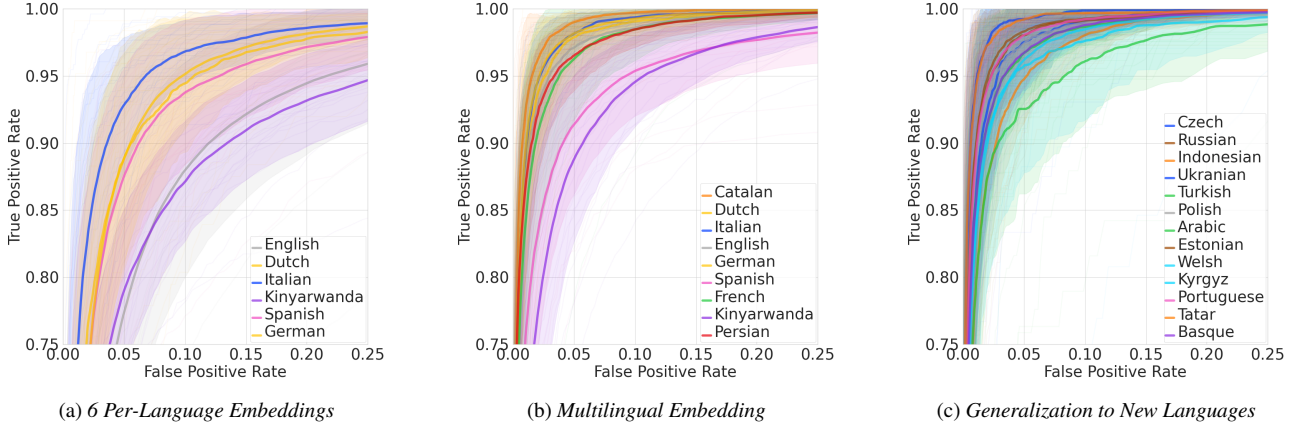
| (a) 6 Per-Language Embeddings | (b) Multilingual Embedding | (c) Generalization to New Languages |

Figure 2: **5-Shot KWS Classification Accuracy**. *ROC curves for 5-shot KWS models with 20 randomly selected keywords per language. For each language, the mean is drawn as a bolded curve over the shaded standard deviation (all keywords are shown as a hairline trace). (a) 5-shot KWS models using an embedding representation trained per language for six languages. [Average $F_1@0.8 = 0.58$] (b) 5-shot models using a multilingual embedding trained on nine languages — accuracy improves relative to (a). [Avg. $F_1@0.8 = 0.75$] (c) 5-shot models using the same multilingual embedding from (b) for random keywords in 13 languages which are out-of-embedding (i.e., which the feature extractor has never encountered), showing that our embedding generalizes to new languages. [Avg. $F_1@0.8 = 0.65$]*
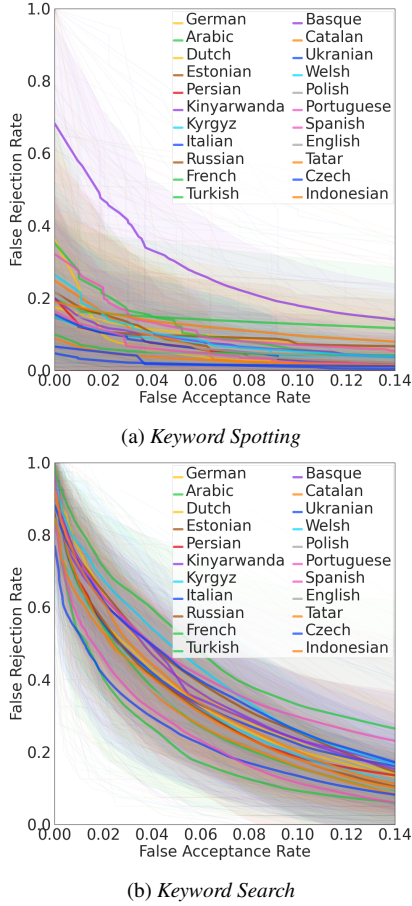

(a) *Keyword Spotting*

(b) *Keyword Search*

Figure 3: **Streaming Accuracy**. *5-shot KWS models evaluated on (a) a stream of target and non-target words emulating a wakeword setting, and (b) keyword search on a stream of spoken sentences. At a threshold of 0.8, (a) achieves an average TPR of 85.2% and FPR of 1.2% over 22 languages, including 13 out-of-embedding languages. (b) shows potential keyword search capabilities, with an average @0.8 TPR of 19% and FPR of 0.2%, despite our system not being optimized for this objective.*

at a threshold of 0.8, suggesting the multilingual embedding model generalizes beyond the languages seen when training it.

5.2. Five-Shot Streaming Accuracy Results

Fig. 3 reports streaming accuracy as the false acceptance rate (FAR) vs. the false rejection rate per instance of true positives (FRR) in each streaming regime (Sec. 4.2). In Fig. 3a, for some keywords, we achieve perfect precision and recall, even in out-of-embedding languages and despite having only one to five speakers in our training set. Fig. 3b depicts our performance on keyword search in full sentences of spoken audio. We reliably find approximately 20%-40% of keywords, despite not optimizing our system for this use-case. We pad all extractions out to 1 second with silence, but including the surrounding audio context when extracting 1 second around each keyword may increase performance in keyword search applications.

6. Conclusions

We demonstrate 5-shot KWS for arbitrary keywords in 22 languages, using an embedding representation pre-trained on an automatically generated dataset. Our embedding generalizes beyond the nine languages it was trained on, and in future work we will investigate support for languages in which word-level alignment is impractical. We utilize a simple fine-tuning scheme for few-shot learning, and continuation studies will explore natural extensions in existing literature (e.g., [23]). We will also pursue a smaller embedding representation via knowledge distillation for on-device deployment.

7. Acknowledgements

8. References

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

[2] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019.

[3] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal forced aligner [computer program]," 2017.

[4] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4087–4091.

[5] 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.

[6] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," *arXiv preprint arXiv:1711.07128*, 2017.

[7] R. Tang and J. Lin, "Deep residual learning for small-footprint keyword spotting," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5484–5488.

[8] C. Banbury, C. Zhou, I. Fedorov, R. M. Navarro, U. Thakker, D. Gope, V. J. Reddi, M. Mattina, and P. N. Whatmough, "Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers," 2021.

[9] G. Chen, C. Parada, and T. N. Sainath, "Query-by-example keyword spotting using long short-term memory networks," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5236–5240.

[10] R. Menon, H. Kamper, E. van der Westhuizen, J. Quinn, and T. Niesler, "Feature Exploration for Almost Zero-Resource ASR-Free Keyword Spotting Using a Multilingual Bottleneck Extractor and Correspondence Autoencoders," in *Proc. Interspeech 2019*, 2019, pp. 3475–3479. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-1665

[11] R. Menon, H. Kamper, J. Quinn, and T. Niesler, "Fast asr-free and almost zero-resource keyword spotting using dtw and cnns for humanitarian monitoring," 2018.

[12] J. Huang, W. Gharbieh, H. S. Shim, and E. Kim, "Query-by-example keyword spotting system using multi-head attention and softtriple loss," 2021.

[13] T. Bluche and T. Gisselbrecht, "Predicting Detection Filters for Small Footprint Open-Vocabulary Keyword Spotting," in *Proc. Interspeech 2020*, 2020, pp. 2552–2556. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2020-1186

[14] N. San, M. Bartelds, M. Browne, L. Clifford, F. Gibson, J. Mansfield, D. Nash, J. Simpson, M. Turpin, M. Vollmer, S. Wilmoth, and D. Jurafsky, "Leveraging neural representations for facilitating access to untranscribed speech from endangered languages," 2021.

[15] J. Lin, K. Kilgour, D. Roblek, and M. Sharifi, "Training keyword spotters with limited and synthesized speech data," 2020.

[16] R. David, J. Duke, A. Jain, V. J. Reddi, N. Jeffries, J. Li, N. Kreeger, I. Nappier, M. Natraj, S. Regev *et al.*, "Tensorflow lite micro: Embedded machine learning on tinyml systems," *arXiv preprint arXiv:2010.08678*, 2020.

[17] TensorFlow, "TensorFlow Micro Frontend," https://github.com/tensorflow/tensorflow/tree/master/tensorflow/lite/experimental/microfrontend, 2021, [Online; accessed 15-Mar-2021].

[18] Keras, "EfficientNet B0," https://keras.io/api/applications/efficientnet/, 2021, [Online; accessed 15-Mar-2021].

[19] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," *Advances in neural information processing systems*, vol. 30, pp. 971–980, 2017.

[20] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 2613–2617. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-2680

[21] TensorFlow, "TinyConv," https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/speech_commands/models.py, 2021, [Online; accessed 15-Mar-2021].

[22] E. Hermann and S. Goldwater, "Multilingual bottleneck features for subword modeling in zero-resource languages," in *Proc. Interspeech 2018*, 2018, pp. 2668–2672. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-2334

[23] J. Snell, K. Swersky, and R. S. Zemel, "Prototypical networks for few-shot learning," 2017.