Low-resource ASR-free keyword spotting using listen-and-confirm

Ewald van der Westhuizen[1], *Marco Ribeiro*[1], *Joshua Jansen van Vüren*[1], *Paula Hidalgo-Sanchis*[2], *Thomas Niesler*[1]

[1]Department of Electrical and Electronic Engineering, Stellenbosch University, South Africa
[2]UN GlobalPulse Lab, Kampala, Uganda

ewaldvdw@sun.ac.za, ribeirompl@sun.ac.za, jjvanvueren@sun.ac.za
hidalgo-sanchis@unglobalpulse.org, trn@sun.ac.za,

Abstract

We present listen-and-confirm (LAC), a human-in-the-loop approach for improving extremely low-resource convolutional neural network-based (CNN) keyword spotting (KWS). LAC interactively presents short audio segments, detected by the KWS, to a human evaluator who confirms whether or not the keyword is present. These LAC responses are used to adjust the CNN training targets and then obtain an improved KWS. Experiments were conducted in English, for controlled experimentation, and Bambara, a severely under-resourced Malian language reflecting the true operational setting in which the KWS is currently used for humanitarian support. Relative improvements in mean precision of 38.18 and 21.62%, respectively, for English and Bambara were achieved for an audio query-by-example task after incorporating feedback from 50 LAC evaluations per keyword type. As a key finding, we show that LAC improves keyword spotting performance even when the human evaluator is completely unfamiliar with the target language. Therefore, LAC can be used to support rapid KWS development in a completely new language.

Index Terms: keyword spotting, human-in-the-loop, under-resourced language

1. Introduction

Community radio phone-in talk shows are often used by the general public to voice matters of pressing social importance. This is especially relevant in countries where social media cannot be used for this purpose, due to poorly developed internet infrastructure. The United Nations (UN) uses keyword spotting systems to monitor such community radio in order to support their relief and developmental programmes in Africa [1, 2]. Currently, automatic speech recognition (ASR) systems[1], which depend on the availability of manually-transcribed speech in the target language, are used for this purpose (left branch of Figure 1). However, for most indigenous African languages such speech resources do not exist and the development thereof is time-consuming, expensive and requires linguistic expertise. In these circumstances, *ASR-free* keyword spotting approaches, that can be developed without substantial labelled data, become attractive [3, 4, 5, 6]. These systems directly detect the presence of a keyword in the speech signal without requiring intermediate lattices or text (right branch of Figure 1).

Dynamic time warping-based (DTW) query-by-example (QbE) is an established approach to ASR-free keyword spotting [7, 8, 9, 10, 11, 12]. DTW performs acoustic matching between a query audio template and a collection of search utterances. Although DTW is straightforward to implement, it is computationally expensive and does not extend easily to large-scale application. Previous work has demonstrated how a convolutional neural network (CNN) can be trained to mimic the behaviour of a DTW-based keyword spotter [11, 12, 13, 14]. The execution speed of this CNN-based approach is three orders of magnitude faster than DTW with a tolerable loss in performance and it allows large-scale, faster than real-time application. However, performance of both DTW and CNN architectures remain constrained by the extremely small training set.

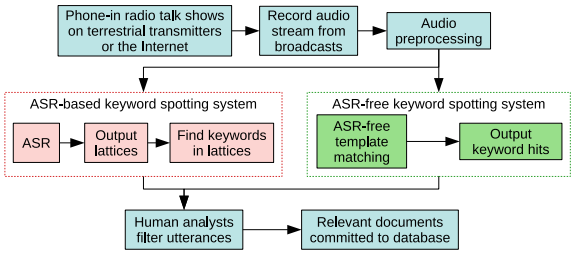

Figure 1: *ASR (red) and ASR-free (green) radio browsing.*

In this paper, we investigate a human-in-the-loop (HITL) strategy for improving the performance of a CNN-based keyword spotter. HITL is an interactive strategy whereby a human provides feedback to a machine learning model by verifying some of its decisions. Research interest in this topic has resurged due to its ability to improve accuracy through minimal time commitment [15]. Our contributions in this paper include: (1) introducing a HITL strategy, which we will refer to as "listen-and-confirm" (LAC), that requires minimal human input; (2) showing that LAC allows keyword spotting performance to be improved even when the human is unfamiliar with the target language; (3) showing that the CNN keyword spotter with LAC outperforms the much more computationally intensive DTW it is trained to mimic.

We describe the datasets in Section 2 and feature extraction in Section 3. The DTW and CNN keyword spotters are described in Section 4, followed by a description of LAC in Section 5. The experimental setup and evaluation metrics are described in Section 6, followed by the results in Section 7. Conclusions and future work are presented in Section 8.

2. Data

Experiments are conducted in South African English and Bambara, a national language of Mali. English is well resourced and therefore allows thorough development experiments to be performed. In contrast, Bambara represents a currently relevant practical application in a truly low-resource setting.

[1]Examples of identified radio broadcasts can be found at https://radio.unglobalpulse.net/uganda.

Table 1: *The training, development and test partitions of the South African English and Bambara datasets. (Utt.: Number of utterances; Dur.: Duration in hours.)*

Set	English		Bambara	
	Utt.	**Dur.**	**Utt.**	**Dur.**
Training	4 220	7.80	11 715	7.45
Development	2 739	5.32	—	—
Test	5 005	10.28	12 790	7.77
Total	11 964	23.40	24 505	15.22

2.1. The English and Bambara in-domain corpora

We use a corpus of South African Broadcast News (SABN), which consists of 23 hours of English speech compiled from news bulletins broadcast between 1996 and 2006 [16]. The corpus contains a mix of newsreader speech, interviews and crossings to reporters. Approximately 80% of the speakers can be considered native English speakers.

The Bambara data was recorded from public radio broadcasts in Mali. First-language speakers manually segmented and transcribed the speech using a Latin script. In comparison to the English data, these recordings are noisy due to FM transmission and telephony compression distortions. This, however, represents the practical setting in which the keyword spotting systems must operate.

The datasets are partitioned into training, development and test sets, as shown in Table 1. For both languages, the training sets are used experimentally as in-domain *untranscribed* data for the computation of DTW targets with which to train the CNN. Hyperparameter optimisation is performed only for the English development data. Thereafter, the approach is applied without further tuning to both the English and Bambara test sets, which fulfil the role of search data. Keyword spotting performance is evaluated using the test set transcriptions. The only situation in which the training set transcriptions are used, is the oracle experiment of Section 6.3. In this experiment, the transcriptions are used to simulate an exhaustive LAC task.

2.2. The English and Bambara keyword corpora

A small, independent corpus of isolated keywords has been compiled in each language. For English, 40 keywords, each uttered twice by 24 South African speakers, were recorded to produce a set of 1920 isolated keyword utterances. This 34-minute corpus of speech represents the only labelled English data used to train the keyword spotting system. There is no speaker overlap with the English in-domain dataset (Section 2.1).

Similarly, 30 Bambara keywords, each uttered twice by 139 Malian speakers, have been recorded to produce a set of 8335 isolated keyword utterances. In our previous work, the elicitation of utterances was prompted by displaying the keywords one at a time on slides [14]. However, for Bambara, utterances were elicited using audio cues instead of text prompts due to the low literacy rate in Mali. The recording conditions are highly variable and often include background noise. This 2-hour set of speech represents the only labelled Bambara data used to train the keyword spotting system. There is no speaker overlap with the Bambara radio dataset (Section 2.1). Only 23 of the 30 keyword types occur ten or more times in the Bambara test set (Table 1). While our keyword spotter is trained to recognise all 30 keyword types, only this subset of 23 keywords is used for test set evaluations in order to ensure reliable evaluation metrics. The composition of the keyword corpora is shown Table 2.

Table 2: *The isolated keyword datasets for English and Bambara, indicating the number of keyword types, speakers, utterances and total duration.*

Language	Keywords	Speakers	Utterances	Duration
English	40	24	1 920	34.27m
Bambara	30	139	8 335	2.05h

These isolated keyword recordings are the only transcribed data used to train our keyword spotters. Furthermore, since data collection is prompted, transcription is implicit and requires no linguistic effort. The idea is that these seed corpora are as easy as possible to collect in a new and challenging environment.

It is worth noting that the keyword frequency in the in-domain corpora (Section 2.1) is extremely low. The keywords with the highest and lowest frequencies occur 645 and 14 times, respectively, in the English corpus, while for the Bambara corpus these numbers are 1099 and zero, respectively. This imbalance between the keyword present (positive) and absent (negative) classes reflects the true operational setting.

3. Feature extraction

Bottleneck features (BNFs) obtained from a neural network trained on well-resourced multilingual speech have been shown to perform well when applied to a variety of tasks in low-resourced languages [17, 18, 19, 20, 21, 22, 23, 24, 25]. We use a multilingual BNF extractor based on the block softmax deep neural network architecture which is implemented using Kaldi [17, 26, 27]. The network is trained using 444 hours of speech data from the nine South African Bantu languages[2] present in the freely-available NCHLT speech corpora [28]. The BNF neural network comprises six 1024-dimensional hidden layers followed by a 39-dimensional linear bottleneck layer and a terminating block softmax output layer. The hidden layers are shared across languages while the block softmax output layer separates the phone state posterior training targets per language. The input features comprise high resolution 40-dimensional MFCCs (no derivatives), 3-dimensional pitch features and 100-dimensional i-vectors for speaker adaptation. The bottleneck layer is used for BNF extraction. Since our preliminary keyword spotting experiments indicated that these multilingual BNFs consistently outperformed MFCCs, they are used in all the experiments that follow.

4. Keyword classifiers

DTW aligns two sequences of feature vectors by warping their time axes to achieve the best match, and requires only a single audio template. The alignment cost associated with this match is a measure of similarity between the sequences. Our baseline DTW keyword spotter determines whether a keyword is present in an utterance by sliding each keyword template over the search utterance and computing the DTW similarity within each window of overlap [14]. During DTW, the cosine frame-wise similarity score is used and the window of overlap corresponds to the length of the template. Since we have multiple templates for each keyword type, the final score indicating whether or not a keyword occurs in an utterance is taken as the highest similarity score over all windows and all templates of that keyword type. To perform keyword spotting, a threshold is applied to this score to decide whether or not a keyword is present.

[2]The languages comprise isiNdebele, Sepedi, Sesotho, siSwati, Setswana, Xitsonga, Tshivenda, isiXhosa and isiZulu.

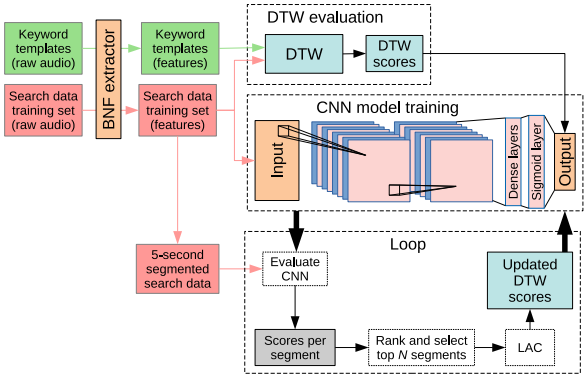

Figure 2: *Diagram showing CNN training using LAC.*

The upper half of Figure 2 illustrates the training and operation of the CNN keyword spotter. DTW similarity scores are calculated between the set of keywords (Section 2.2) and the untranscribed training data (Section 2.1). These scores are used to train a CNN. Note that this does not correspond to fully-supervised end-to-end (E2E) keyword spotting, also described in [14]. The approach used here has been shown to outperform E2E keyword spotting in the low-resource setting we describe.

5. Listen-and-confirm (LAC)

The lower half of Figure 2 illustrates CNN keyword spotting using the proposed LAC procedure. First, the CNN keyword spotter is trained as described in Section 6.2. The training set utterances are then split into five-second segments, with a two-second overlap to provide shorter audio samples that are easy for a human to assess quickly. The CNN is used to evaluate these segments, after which they are ranked according to the assigned scores. A top N selection of these training segments is then presented to a human listener, who is asked to confirm whether the keyword under consideration does indeed occur in the audio segment. These answers are used to adjust the initially-obtained DTW training target scores. The pseudocode in Algorithm 1 describes how these scores are adjusted. The answers from the LAC task are used to identify training utterances that contain confirmed keywords. The target scores for these utterances are set to one. The training set, including these adjustments, is then used to train a new CNN model.

Preliminary experiments used both confirmation and rejection answers from the LAC task to adjust the DTW scores. However, this approach yielded consistently degraded performance. We believe that this degradation occurs because DTW aims to find the best match of a template in a search segment, and consequently the CNN is trained to mimic this behaviour. Assigning a decreased score as a training target to indicate an acoustic mismatch therefore contradicts what DTW as a sequence matching algorithm tries to achieve and is counterproductive. Therefore, we have included only confirmation LAC answers during retraining.

6. Experiments

PyTorch and an open source implementation of DTW[3] were used to conduct all experiments. Precision-recall curves (PRC), which plot classifier precision against the recall as the detection threshold is varied, provide an informative reflection of model performance on imbalanced datasets such as ours [29, 30]. We therefore calculate the average precision (AP) for each keyword type, which is the area under the PRC, as a performance metric. For multilabel classification, the mean average precision (mAP), which is the mean AP over all keyword types, is used as a single metric that characterises overall classifier performance. We also consider the precision at 10 (P@10) which is the proportion of correct keyword detections among the top 10 detections. The reported P@10 is the mean over all keyword types.

[3] https://github.com/kamperh/speech_dtw

Algorithm 1 Pseudocode describing the adjustment of DTW scores in response to LAC.

define $\text{lo}(x)$: Length of array x
Input: U: Array of search utterances
Input: K: Array of keyword type labels
Input/Output: S: Two-dimensional array of aggregated DTW scores; size: $[\text{lo}(U), \text{lo}(K)]$
Scale the values of S so that 1.0 corresponds to the the global maximum (best match) and 0.0 to global minimum (poorest match) in S.
for $u = 1$ **to** $\text{lo}(U)$ **do**
  for $k = 1$ **to** $\text{lo}(K)$ **do**
    if human or oracle confirmed that $U(u)$ contains keyword type $K(k)$ **then**
      $S(u)(k) \leftarrow 1.0$
    end if
  end for
end for
Save the updated S so that it can be used for CNN training.

6.1. DTW baseline

We use a DTW keyword spotter as a first baseline, evaluating its performance on the test sets in Table 1. Furthermore, the DTW scores obtained by the same system on the training sets in Table 1 are used as training targets for the CNN keyword spotter described in Sections 6.2 and 6.3.

6.2. CNN without LAC

As a second baseline, we use a CNN keyword spotter without LAC. The CNN receives a 39×1500 (features×time) tensor as input, corresponding to a 15-second audio signal. Shorter utterances are zero-padded, while longer ones are truncated. The final output layer is dense with sigmoid activation functions and as many units as keyword types in the respective datasets. Thus the model can be considered a collection of binary classifiers, one for each keyword, with shared input layers. Architectural choices have been optimised for performance using the English development set (Table 1). The network is trained for 100 epochs using the summed cross-entropy loss. No transcriptions are used during either training or validation. A more detailed description of the network architecture is presented in [14].

6.3. CNN with LAC

CNN with LAC was performed for 5, 10, 20 and 50 LAC evaluations per keyword type. LAC for both English and Bambara was performed by the same first language English speaker who was unfamiliar with Bambara.

An oracle experiment was also conducted, in which an automated routine instead of a human determines whether or not a keyword occurs in an utterance based on the transcriptions. This was the only situation in which the training set transcriptions were used to provide supervision. The transcriptions represent

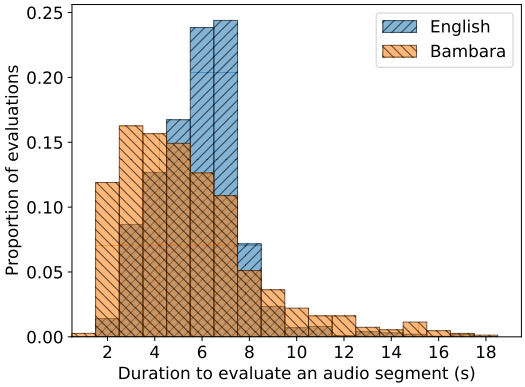

Figure 3: *Distribution of the time required to evaluate a 5-second audio segment during LAC.*

Table 3: *Test set results for DTW and CNN systems. The mAP and P@10 values are shown as percentages. (N/A: Not applicable; #LACs: Number of LAC evaluations per keyword.)*

System	#LACs	English		Bambara	
		mAP	P@10	mAP	P@10
DTW (baseline)	N/A	23.32	39.25	42.87	60.43
CNN without LAC	0	22.29	38.00	39.36	56.96
CNN with LAC	5	23.24	40.75	42.53	60.00
	10	25.14	44.50	44.67	64.35
	20	28.35	46.00	45.75	63.91
	50	30.80	49.25	47.87	66.52
CNN oracle	All	62.48	78.25	60.57	79.57

an infallible human listener, able to consider the entire training set, and therefore allow us to assess the maximum improvement that can be achieved by LAC on the datasets in Section 2.1.

7. Results

We present an analysis of the time spent by the human evaluator performing LAC, followed by the performance of the different keyword spotting systems described in Section 6.

Figure 3 presents histograms of the time required to evaluate 2 000 English and 1 500 Bambara audio segments during LAC by the same evaluator. This corresponds to 50 LAC evaluations per keyword type per language. Although the evaluator is proficient in English and has no knowledge of Bambara, the distributions show a high degree of overlap, indicating that both tasks were performed at roughly the same pace. In total, 3.57 and 2.49 hours respectively were spent performing English and Bambara LAC, while the mean and standard deviation of the time required to evaluate one 5-second audio segment was 6.43 ± 1.94 and 5.99 ± 3.09 seconds respectively. The lower standard deviation for English indicates a more regular pace, possibly due to the proficiency of the evaluator. This might imply that the task was considered more difficult in the unfamiliar language. Despite this, LAC did not on average require more time per audio segment for the unfamiliar language.

Table 3 shows the mAP and P@10 results for the experiments described in Section 6. The CNN with five LAC evaluations per keyword type, which require roughly 20 minutes in total of human effort, closes the gap observed between DTW and the CNN without LAC. Applying more than five LAC evaluations yields consistent further improvements. The CNN word spotters trained with 50 LAC evaluations per keyword type yield a 38.18 and 21.62% relative improvements in mAP for English and Bambara respectively, compared to the CNN without LAC. Similar improvements are observed for P@10. The oracle systems, which simulate perfect LAC evaluations using the entire training sets, indicate the potential for further improvement.

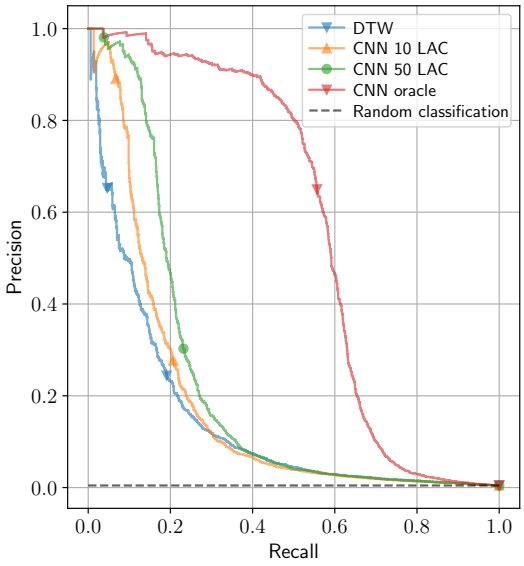

Figure 4: *Test set precision-recall curves for different Bambara keyword spotters.*

8. Conclusions

We have presented and evaluated "listen-and-confirm" (LAC), a human-in-the-loop approach to improving the performance of a CNN-based ASR-free keyword spotting system in an extremely low-resource scenario, where only a few isolated keyword templates and a larger corpus of in-domain, but untranscribed, speech are available. Experiments were conducted for English, a well-resourced language, and Bambara, a low-resource language which reflects a current true operational setting in which the keyword spotting system is used to support humanitarian relief efforts. Our baseline CNN-based keyword spotter, trained on automatically-obtained training targets provided by DTW, yields a mAP of 22.29 and 39.36% for English and Bambara respectively. By including LAC feedback during CNN re-training, substantial and consistent improvements in the mAP and P@10 metrics were achieved. A key finding is that this was possible even when the human evaluator was completely unfamiliar with the target language. Therefore, LAC can support the rapid development of keyword spotting systems in new and severely under-resourced languages, where annotation of speech data may be impossible due to lacking linguistic expertise and time constraints. Future work will focus on implementing an iterative approach for model updates during LAC.

9. Acknowledgements

We gratefully acknowledge NVIDIA corporation for the donation of GPU equipment, the South African Council for Scientific and Industrial Research and Department of Science and Technology for use of the Lengau CHPC cluster, and the support of Telkom South Africa.

10. References

[1] R. Menon, A. Saeb, H. Cameron, W. Kibira, J. Quinn, and T. Niesler, "Radio-browsing for developmental monitoring in Uganda," in *Proc. ICASSP*, 2017.

[2] A. Saeb, R. Menon, H. Cameron, W. Kibira, J. Quinn, and T. Niesler, "Very low resource radio browsing for agile developmental and humanitarian monitoring," in *Proc. Interspeech*, 2017.

[3] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks." in *Proc. ICASSP*, 2014.

[4] T. N. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," in *Proc. Interspeech*, 2015.

[5] K. Audhkhasi, A. Rosenberg, A. Sethy, B. Ramabhadran, and B. Kingsbury, "End-to-end ASR-free keyword search from speech," in *Proc. ICASSP*, 2017.

[6] R. Tang and J. Lin, "Deep residual learning for small-footprint keyword spotting," in *Proc. ICASSP*, 2018.

[7] A. S. Park and J. R. Glass, "Unsupervised pattern discovery in speech," *IEEE Trans. Audio, Speech, Language Process.*, vol. 16, no. 1, pp. 186–197, 2008.

[8] T. J. Hazen, W. Shen, and C. White, "Query-by-example spoken term detection using phonetic posteriorgram templates," in *Proc. ASRU*, 2009.

[9] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on gaussian posteriorgrams," in *Proc. ASRU*, 2009.

[10] A. Jansen and B. Van Durme, "Indexing raw acoustic features for scalable zero resource search," in *Proc. Interspeech*, 2012.

[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," in *Proc. Interspeech*, 2018.

[12] ——, "ASR-free CNN-DTW keyword spotting using multilingual bottleneck features for almost zero-resource languages," in *Proc. SLTU*, 2018.

[13] 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.

[14] E. van der Westhuizen, H. Kamper, R. Menon, J. Quinn, and T. Niesler, "Feature learning for efficient ASR-free keyword spotting in low-resource languages," *Computer Speech and Language*, vol. 71, 2022.

[15] X. Wu, L. Xiao, Y. Sun, J. Zhang, T. Ma, and L. He, "A survey of human-in-the-loop for machine learning," *CoRR*, vol. abs/2108.00941, 2021. [Online]. Available: https://arxiv.org/abs/2108.00941

[16] H. Kamper, F. De Wet, T. Hain, and T. Niesler, "Capitalising on North American speech resources for the development of a South African English large vocabulary speech recognition system," *Computer Speech and Language*, vol. 28, no. 6, pp. 1255–1268, 2014.

[17] K. Veselỳ, M. Karafiát, F. Grézl, M. Janda, and E. Egorova, "The language-independent bottleneck features," in *Proc. SLT*, 2012.

[18] N. T. Vu, W. Breiter, F. Metze, and T. Schultz, "An investigation on initialization schemes for multilayer perceptron training using multilingual data and their effect on ASR performance," in *Proc. Interspeech*, 2012.

[19] J. Cui *et al.*, "Multilingual representations for low resource speech recognition and keyword search," in *Proc. ASRU*, 2015.

[20] T. Alumäe, S. Tsakalidis, and R. M. Schwartz, "Improved multilingual training of stacked neural network acoustic models for low resource languages," in *Proc. Interspeech*, 2016.

[21] S. Thomas, S. Ganapathy, and H. Hermansky, "Multilingual MLP features for low-resource LVCSR systems," in *Proc. ICASSP*, 2012.

[22] H. Chen, C.-C. Leung, L. Xie, B. Ma, and H. Li, "Multilingual bottle-neck feature learning from untranscribed speech," in *Proc. ASRU*, 2017.

[23] Y. Yuan, C.-C. Leung, L. Xie, H. Chen, B. Ma, and H. Li, "Extracting bottleneck features and word-like pairs from untranscribed speech for feature representation," in *Proc. ASRU*, 2017.

[24] E. Hermann and S. Goldwater, "Multilingual bottleneck features for subword modeling in zero-resource languages," in *Proc. Interspeech*, 2018.

[25] E. Hermann, H. Kamper, and S. J. Goldwater, "Multilingual and unsupervised subword modeling for zero resource languages," *Computer Speech and Language*, vol. 65, 2021.

[26] R. Fer, P. Matějka, F. Grézl, O. Plchot, K. Veselỳ, and J. H. Černockỳ, "Multilingually trained bottleneck features in spoken language recognition," *Computer Speech & Language*, vol. 46, pp. 252–267, 2017.

[27] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz, J. Silovsk, G. Stemmer, and K. Veselý, "The Kaldi speech recognition toolkit," in *Proc. ASRU*, 2011.

[28] E. Barnard, M. H. Davel, C. van Heerden, F. De Wet, and J. Badenhorst, "The NCHLT speech corpus of the South African languages," in *Proc. SLTU*, St. Petersburg, Russia, 2014.

[29] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. ICML*, 2006.

[30] T. Saito and M. Rehmsmeier, "The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets," *PLoS ONE*, vol. 10, 2015.