IberSPEECH 2018
21-23 November 2018, Barcelona, Spain

GTTS-EHU Systems for the Albayzin 2018 Search on Speech Evaluation

Luis Javier Rodríguez-Fuentes, Mikel Peñagarikano, Aparo Varona, Germán Bordel

Departamento de Electricidad y Electrónica UPV/EHU
Barrio Sarriena s/n, 48940 Leioa, Bizkaia, Spain
luisjavier.rodriguez@ehu.eus

Abstract

This paper describes the systems developed by GTTS-EHU for the QbE-STD and STD tasks of the Albayzin 2018 Search on Speech Evaluation. Stacked bottleneck features (sBNF) are used as frame-level acoustic representation for both audio documents and spoken queries. In QbE-STD, a flavour of segmental DTW (originally developed for MediaEval 2013) is used to perform the search, which iteratively finds the match that minimizes the average distance between two test-normalized sBNF vectors, until either a maximum number of hits is obtained or the score does not attain a given threshold. The STD task is performed by synthesizing spoken queries (using publicly available TTS APIs), then averaging their sBNF representations and using the average query for QbE-STD. A publicly available toolkit (developed by BUT/Phonexia) has been used to extract three sBNF sets, trained for English monophone and triphone state posteriors (contrastive systems 3 and 4) and for multilingual triphone posteriors (contrastive system 2), respectively. The concatenation of the three sBNF sets has been also tested (contrastive system 1). The primary system consists of a discriminative fusion of the four contrastive systems. Detection scores are normalized on a query-by-query basis (*qnorm*), calibrated and, if two or more systems are considered, fused with other scores. Calibration and fusion parameters are discriminatively estimated using the ground truth of development data. Finally, due to a lack of robustness in calibration, Yes/No decisions are made by applying the MTWV thresholds obtained for the development sets, except for the COREMAH test set. In this case, calibration is based on the MAVIR corpus, and the 15% highest scores are taken as positive (Yes) detections.

Index Terms: Spoken Term Detection, Query-by-Example Spoken Term Detection, Bottleneck features, Dynamic Time Warping

1. Introduction

The main goal of the participation of GTTS in the Albayzin 2018 Search on Speech Evaluation was to upgrade the QbE-STD systems that we developed for MediaEval 2013 and 2014 [1] [2] by: (1) using an external VAD module; (2) replacing phonetic posteriors by bottleneck features as frame-level features (which might imply changing some aspects of the methodology); and (3) handling the case of no development data by applying cross-condition calibration and heuristic thresholding. Also, as a proof-of-concept attempt to overcome the issue of OOV words, we have developed STD systems by applying public TTS APIs to synthesize a number of spoken instances of each term, then computing an average query (following the approach developed for MediaEval 2013 [3]) and using it to perform QbE-STD.

QbE-STD results on development data reveal that calibration models are not well estimated, probably due to a lack of detections for target trials. In particular, the threshold given by the application model parameters ($P_{target} = 0.0001$, $C_{miss} = 1$ and $C_{fa} = 0.1$) is too high compared to the MTWV threshold.

In the case of STD, the lack of detections is even more remarkable, with $P_{miss} \approx 0.8$ for extremely low thresholds. This could be due to an acoustic mismatch between the synthesized queries and the test audio signals, which might be blocking the DTW-based search, since the score yielded by the best match would fall below the established threshold. The low amount of detections (especially, for target trials) not only yields high miss error rates but also makes the calibration model to be poorly estimated. This may explain why the MTWV threshold for STD scores on development data is much lower that that provided by the application model.

In both cases (QbE-STD and STD), the parameters of the DTW-based search should be further tuned in order to get a larger amount of target and non-target detections.

2. QbE-STD systems

In the design of our previous QbE-STD systems, the front-end exploited existing software (e.g. the BUT phone decoders for Czech, Hungarian and Russian [4]), whereas the backend (search, calibration and fusion) was almost entirely developed by our group (and collaborators) [3][5]. For this evaluation, we have applied an external VAD module and extracted a new set of frame-level features.

2.1. Voice Activity Detection (VAD)

In our previous QbE-STD systems, VAD was performed by using the posteriors provided BUT phone decoders: first, the posteriors of non-phonetic units were added; then, if this aggregated non-phonetic posterior was higher than any other (phonetic) posterior, the frame was labeled as *non-speech*; otherwise it was labeled as *speech*. In this evaluation, we are not using posteriors anymore, so we cannot apply the same procedure. Instead, we apply the Python interface to a VAD module developed by Google for the WebRTC project [6], based on Gaussian distributions of speech and non-speech features. Given an audio file, our VAD module produces two output files: the first one (required by the feature extraction module) is an HTK .lab file specifying speech and non-speech segments, whereas the second one (used by the search procedure) is a text file (.txt) containing a sequence of 1's and 0's (one per line) indicating speech/non-speech frames.

2.2. Bottleneck features

We have updated the feature extraction module of our previous QbE-STD systems with the stacked botteneck features (sBNF) recently presented by BUT/Phonexia [7]. Actually, three different neural networks are applied, each one trained to classify a different set of acoustic units and later optimized to language

10.21437/IberSPEECH.2018-52

recognition tasks. The first network was trained on telephone speech (8 kHz) from the English Fisher corpus [8] with 120 monophone state targets (FisherMono); the second one was also trained on the Fisher corpus but with 2423 triphone tied-state targets (FisherTri); the third network was trained on telephone speech (8 kHz) in 17 languages taken from the IARPA Babel program [9], with 3096 stacked monophone state targets for the 17 languages involved (BabelMulti).

The architecture of these networks consists of two stages. The first one is a standard bottleneck network fed with low-level acoustic features spanning 10 frames (100 ms), the bottleneck size being 80. The second stage takes as input five equally spaced BNFs of the first stage, spanning 31 frames (310 ms), and is trained on the same targets as the first stage, with the same bottleneck size (80). The bottleneck features extracted from the second stage are known as stacked bottleneck features (sBNF). Alternatively, instead of sBNF, the extractor can output target posteriors.

The operation of BUT/Phonexia sBNF extractors requires an external VAD module providing speech/non-speech information through an HTK .lab file. If no external VAD is provided, a simple energy-based VAD is computed internally. In this evaluation, we have applied the WebRTC VAD described above.

Our first aim was to replace old BUT by new BUT/Phonexia posteriors, but the huge size of FisherTri (2423) and BabelMulti (3096) targets required some kind of selection, clustering or dimensionality reduction approach. So, given that —at least theoretically— the same information is conveyed by sBNF's, with a suitably low dimensionality (80), we decided to switch from posteriors to sBNF's. We were aware that this change may make us pay a high price. Posteriors have a clear meaning, they can be linearly combined and their values suitably fall within the range [0,1], which makes the $-\log\cos(\alpha)$ distance also range in [0,1] (α being the angle between two vectors of posteriors), with very good results reported in our previous works. On the other hand, bottleneck layer activations have no clear meaning, we don't really know if they can be linearly combined (e.g for computing an average query from multiple query instances), and their values are unbounded, so the $-\log\cos(\alpha)$ distance does no longer apply. Is there any other distance working fine with sBNF? This evaluation poses a great opportunity to address these issues.

2.3. DTW-based search

To perform the search of spoken queries in audio documents, we basically follow the DTW-based approach presented in [3]. In the following, we summarize the approach and the modifications introduced for this evaluation.

Given two sequences of sBNF's corresponding to a spoken query and an audio document, we first apply VAD to discard non-speech frames, but keeping the timestamp of each frame. To avoid memory issues, audio documents are splitted into chunks of 5 minutes, overlapped 5 seconds, and processed independently. This chunking process is key to the speed and feasibility of the search procedure.

Let us consider the VAD-filtered sequences corresponding to a query $q = (q[1], q[2], \ldots, q[m])$ and an audio document $x = (x[1], x[2], \ldots, x[n])$, of length m and n, respectively. Since sBNF's (theoretically) range from $-\infty$ to $+\infty$, we define the distance between any pair of vectors, $q[i]$ and $x[j]$, as follows:

$$d(q[i], x[j]) = -\log\left(1 + \frac{q[i] \cdot x[j]}{|q[i]| \cdot |x[j]|}\right) + \log 2 \quad (1)$$

Note that $d(v, w) \geq 0$, with $d(v, w) = 0$ if and only if v and w are aligned and pointing in the same direction, and $d(v, w) = +\infty$ if and only if v and w are aligned and pointing in opposite directions.

The distance matrix computed according to Eq. 1 is normalized with regard to the audio document x, as follows:

$$d_{norm}(q[i], x[j]) = \frac{d(q[i], x[j]) - d_{min}(i)}{d_{max}(i) - d_{min}(i)} \quad (2)$$

where:

$$d_{min}(i) = \min_{j=1,\ldots,n} d(q[i], x[j]) \quad (3)$$

$$d_{max}(i) = \max_{j=1,\ldots,n} d(q[i], x[j]) \quad (4)$$

In this way, matrix values are in the range $[0, 1]$ and a perfect match would produce a quasi-diagonal sequence of zeroes. This can be seen as *test nomalization* since, given a query q, distance matrices take values in the same range (and with the same *relative meaning*), no matter the acoustic conditions, the speaker, etc. of the audio document x.

Note that the chunking process described above makes the normalization procedure differ from that applied in [3], since $d_{min}(i)$ and $d_{max}(i)$ are not computed for the whole audio document but for each chunk independently. On the other hand, considering chunks of 5 minutes might be beneficial, since normalization is performed in a more local fashion, that is, more suited to the speaker(s) and acoustic conditions of each particular chunk.

The best match of a query q of length m in an audio document x of length n is defined as that minimizing the average distance in a *crossing path* of the matrix d_{norm}. A crossing path starts at any given frame of x, $k_1 \in [1, n]$, then traverses a region of x which is optimally aligned to q (involving L vector alignments), and ends at frame $k_2 \in [k_1, n]$. The average distance in this crossing path is:

$$d_{avg}(q, x) = \frac{1}{L} \sum_{l=1}^{L} d_{norm}(q[i_l], x[j_l]) \quad (5)$$

where i_l and j_l are the indices of the vectors of q and x in the alignment l, for $l = 1, 2, \ldots, L$. Note that $i_1 = 1$, $i_L = m$, $j_1 = k_1$ and $j_L = k_2$. The optimization procedure is $\Theta(n \cdot m \cdot d)$ in time (d: size of feature vectors) and $\Theta(n \cdot m)$ in space. For details, we refer to [3].

The detection score is computed as $1 - d_{avg}(q, x)$, thus ranging from 0 to 1, being 1 only for a perfect match. The starting time and the duration of each detection are obtained by retrieving the time offsets corresponding to frames k_1 and k_2 in the VAD-filtered audio document.

This procedure is iteratively applied to find not only the best match but also less likely matches in the same audio document. To that end, a queue of search intervals is defined and initialized with $[1, n]$. Let us consider an interval $[a, b]$, and assume that the best match is found at $[a', b']$, then the intervals $[a, a' - 1]$ and $[b' + 1, b]$ are added to the queue (for further processing) only if the following conditions are satisfied: (1) the score of the current match is greater than a given threshold T (in this evaluation, $T = 0.85$); (2) the interval is long enough (in this evaluation, half the query length: $m/2$); and (3) the number of matches (those already found + those waiting in the queue) is less than a given threshold M (in this evaluation, $M = 7$). An example is shown in Figure 1. Finally, the list of matches for each query is ranked according to the scores and truncated to the N highest scores (in this evaluation, $N = 1000$, though it effectively applied only in a few cases).

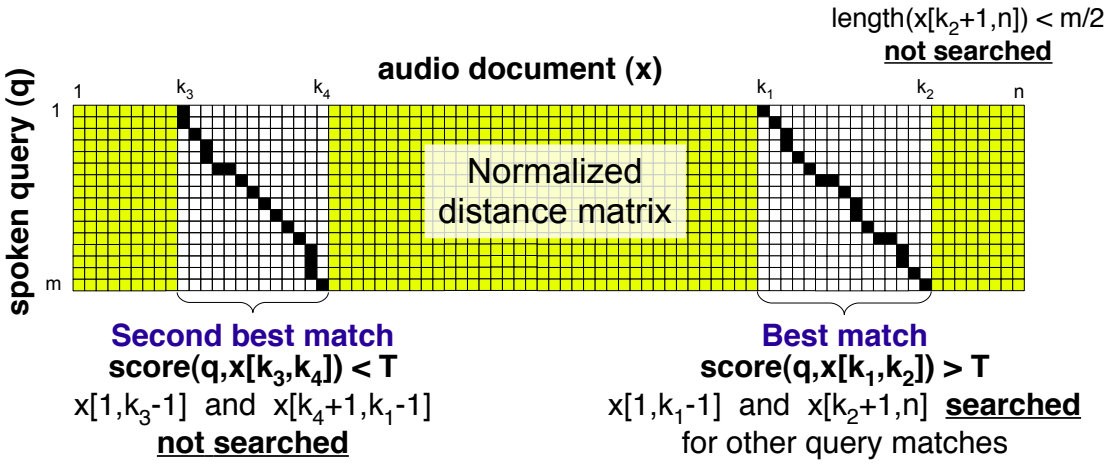

Figure 1: *Example of the iterative DTW procedure: (1) the best match of q in $x[1,n]$ is located in $x[k_1,k_2]$; (2) since the score is greater than the established threshold T, the search continues in the surrounding segments $x[1,k_1-1]$ and $x[k_2+1,n]$; (3) $x[k_2+1,n]$ is not searched, because it is too short; (4) the best match of q in $x[1,k_1-1]$ is located in $x[k_3,k_4]$; (5) but its score is lower than T, so the surrounding segments $x[1,k_3-1]$ and $x[k_4+1,k_1-1]$ are not searched. The search procedure outputs the segments $x[k_1,k_2]$ and $x[k_3,k_4]$.*

2.4. Calibration and fusion of system scores

The scores produced by our systems are transformed according to a discriminative calibration/fusion approach commonly applied in speaker and language recognition, that we adapted to STD tasks for MediaEval 2013, in collaboration with Alberto Abad, from L^{2}F, the Spoken Language Systems Laboratory, INESC-ID Lisboa. In the following paragraphs, we just summarize the procedure. For further details, see [5].

First, the so-called *q-norm* (query normalization) is applied, so that zero-mean and unit-variance scores are obtained per query. Then, if n different systems are fused, detections are aligned so that only those supported by k or more systems ($1 \leq k \leq n$) are retained for further processing (in this evaluation, we use $k = 2$). To build the full set of trials (potential detections) we assume a rate of 1 trial per second (which is consistent with the evaluation script). Now, let us consider one of those detections of a query q supported by at least k systems, and a system A that did not provide a score for it. There could be different ways to fill up this *hole*. We use the minimum score that A has output for query q in other trials. In fact, the minimum score for the query q is hypothesized for all target and non-target trials of query q for which system A has not output a detection score. When a single system is considered ($n = 1$), the majority voting scheme is skipped but *qnorm* and the filling up of missing scores are still applied. In this way, a complete set of scores is prepared, which besides the ground truth (target/non-target labels) for a development set of queries, can be used to discriminatively estimate a linear transformation that will hopefully produce well-calibrated scores.

The calibration/fusion model is estimated on the development set and then applied to both the development and test sets, using the BOSARIS toolkit [10][11]. Under this approach, the Bayes optimal threshold, given the effective prior (in this evaluation, $\hat{P}_{target} = C_{miss}P_{target}/(C_{miss}P_{target} + C_{fa}(1 - P_{target})) = 0.001$), would be applied and —at least theoretically— no further tunings would be necessary. In practice, however, if a system yielded a small amount of detections, we would be using hypothesized scores for most of the trials. As a result, the calibration/fusion model would be poorly estimated and the Bayes optimal threshold (in this evaluation, 6.9) would not produce good results.

3. STD systems

In this evaluation, we have exploited some publicly available Text-to-Speech (TTS) API's to perform text-in-audio search as audio-in-audio search. This is just a proof-of-concept aimed at overcoming the Out-Of-Vocabulary (OOV) word issue.

We have applied the Google TTS (gTTS) Python library and command-line interface (CLI) tool [12], which provides two different female (es-ES and es-US) voices, and the Cocoa interface to speech synthesis in MacOS [13], which provides 5 different voices (three male, two female) including both European and American Spanish.

In this way, for each textual term, we synthesize 7 spoken queries: $q_1, q_2, \ldots, q_7$. These spoken queries are downsampled to 8 kHz and applied VAD and sBNF extraction as described in Sections 2.1 and 2.2. The longest query is then taken as reference and optimally aligned to the other queries by means of a standard DTW procedure. Let us consider the sequence of VAD-filtered sBNF vectors for the reference query: q_l of length m_l, and the sequence corresponding to another synthesized query: q_i of length m_i. The alignment starts at $[1,1]$ and ends at $[m_l, m_i]$ and involves L alignments, such that each feature vector of q_l is aligned to a sequence of vectors of q_i. This is repeated for all the synthesized queries, such that we end up with a set of feature vectors S_j aligned to each feature vector $q_l[j]$, for $j = 1, 2, \ldots, m_l$. Then, each $q_l[j]$ is averaged with the feature vectors in S_j to get a *single average query*, as follows:

$$q_{\text{avg}}[j] = \frac{1}{1 + |S_j|}\left(q_l[j] + \sum_{v \in S_j} v\right) \quad j = 1, 2, \ldots, m_l \tag{6}$$

Finally, the average query obtained in this way is used to search for occurrences in the audio documents, just in the same way (using the same configuration) as we do in the QbE-STD task.

4. Experimental setup and results

Since BUT/Phonexia sBNF extractors operate on 8 kHz signals, all the query and test audio signals have been downsampled to 8 kHz and stored as 16 bit little-endian signed-integer single-channel WAV files. Audio conversion was performed under MacOS, using the following command:

```
afconvert audio.<ext> -o audio_8k.wav
              -d LEI16@8000 -c 1 -f WAVE
```

where <ext> represents any audio format extension (such as mp3, aac, wav, etc.).

Two different datasets have been provided for training and development of QbE-STD and STD systems in the Albayzin 2018 Search on Speech Evaluation [14]: MAVIR [15] and RTVE [16]. We have not used the training dataset at all, nor the dev1 set of RTVE. Only the dev set of MAVIR and the dev2 set of RTVE have been used to estimate the calibration/fusion models, and later to search for query occurrences. Search has been also performed on the test sets: COREMAH [17], MAVIR and RTVE, applying the calibration/fusion models and the optimal (MTWV) thresholds obtained on development. In the case of COREMAH (for which no development data was provided), MAVIR calibration/fusion models have been used and heuristic thresholding has been applied, by making *Yes* decisions for the 15% of detections with the highest scores.

One primary (pri) and four contrastive (con1, con2, con3 and con4) systems have ben submitted to each combination of task (QbE-STD, STD), condition (development, test) and dataset (COREMAH, MAVIR, RTVE). BabelMulti, FisherMono and FisherTri sBNF's were used for contrastive systems 2, 3 and 4, respectively. The concatenation of the three sBNF's (with $80 \times 3 = 240$ dimensions) was used as acoustic representation for contrastive system 1. Finally, the primary system was obtained as the discriminative fusion of the four contrastive systems.

Tables 1 and 2 show ATWV/MTWV performance of the 5 GTTS-EHU systems on the development sets of MAVIR and RTVE for the QbE-STD and STD tasks, respectively. In all cases, ATWV is obtained for the Bayes optimal threshold (6.9). Along with the MTWV score, the MTWV threshold is shown too (in parentheses). As noted above, the systems eventually submitted for MAVIR and RTVE (in all tasks and conditions) were applied the MTWV threshold.

Table 1: *ATWV/MTWV performance on the development sets of MAVIR and RTVE for the QbE-STD systems submitted by GTTS-EHU. The ATWV threshold is set to* 6.9*. Along with the MTWV score, the MTWV threshold is shown in parentheses.*

	MAVIR		RTVE	
	MTWV (Thr)	ATWV	MTWV (Thr)	ATWV
con2	0.1291 (4.06)	0.0000	0.4893 (4.75)	0.0097
con3	0.1327 (4.12)	0.0000	0.5722 (5.14)	0.0802
con4	0.1590 (4.09)	0.0000	0.5227 (5.09)	0.0437
con1	0.1278 (4.14)	0.0000	0.5159 (5.13)	0.0421
pri	0.1577 (4.59)	0.0000	0.5352 (5.69)	0.3043

Table 2: *ATWV/MTWV performance on the development sets of MAVIR and RTVE for the STD systems submitted by GTTS-EHU. The ATWV threshold is set to* 6.9*. Along with the MTWV score, the MTWV threshold is shown in parentheses.*

	MAVIR		RTVE	
	MTWV (Thr)	ATWV	MTWV (Thr)	ATWV
con2	0.0396 (5.12)	0.0000	0.0933 (5.30)	0.0026
con3	0.0463 (4.82)	0.0000	0.0951 (4.80)	0.0000
con4	0.0512 (4.31)	0.0000	0.0916 (4.98)	0.0000
con1	0.0398 (5.28)	0.0000	0.0843 (5.21)	0.0000
pri	0.0464 (5.43)	0.0000	0.0809 (5.91)	0.0265

5. Conclusions and future work

The QbE-STD and STD results obtained by our systems on the development sets of MAVIR and RTVE may indicate that the search procedure is detecting few occurrences of the queries, yielding high miss error rates and making it difficult the estimation of good (robust) calibration/fusion models, since most of the trials are missing from system output and we have to hypothesize scores for them. To get a larger amount of target and non-target detections, three of the parameters of the DTW-based search must be relaxed: the maximum amount M of hits per audio chunk (currently, $M = 7$), the minimum score T required to keep searching (currently, $T = 0.85$) and the maximum number N of detections per query for the whole set of audios (currently, $N = 1000$).

QbE-STD detection scores, though badly calibrated, seem to work fine for RTVE but not so well for MAVIR. The difference in performance might be related to a higher acoustic variability or more adverse conditions (reverberation, noise, etc.) for MAVIR.

The trick of synthesizing spoken queries from textual terms to perform STD as QbE-STD seems to be failing, with two possible causes: (1) an acoustic mismatch between the synthesized queries and the test audios might lead to low scores and block the iterative DTW detection procedure; and (2) the use of bottleneck layer activations as frame-level acoustic representation might be incompatible with the query averaging procedure (which worked fine with phone posteriors).

Future developments may involve some sort of data augmentation in QbE-STD, such as the use of pseudo-relevance feedback, that is, the use of top matching query occurrences as additional examples. Also, though query averaging is computationally cheap, using it with sBNF representations might be unfeasible and other more expensive strategies to take advantage of the synthesized queries should be explored, such as carrying out multiple searches and fusing the results [18]. Alternatively, we could return to posteriors, by combining the high-dimensional sets of posteriors provided by BUT/Phonexia networks with some sort of feature clustering or feature selection approach.

6. Acknowledgements

We thank Javier Tejedor and Doroteo T. Toledano for organizing a new edition of the Search on Speech Evalution and for their help during the development process. We also thank Eduardo Lleida and the ViVoLab team for the huge effort of collecting and annotating RTVE broadcasts for the ALBAYZIN 2018 evaluations. This work has been partially funded by the UPV/EHU under grant GIU16/68.

7. References

[1] X. Anguera, L. J. Rodriguez-Fuentes, F. Metze, I. Szöke, A. Buzo, and M. Penagarikano, "Query-by-example spoken term detection on multilingual unconstrained speech," in *Interspeech 2014*, Singapore, September 14-18 2014, pp. 2459–2463.

[2] X. Anguera, L. J. Rodriguez-Fuentes, A. Buzo, F. Metze, I. Szöke, and M. Penagarikano, "Quesst2014: Evaluating query-by-example speech search in a zero-resource setting with real-life queries," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2015)*, Brisbane, Australia, April 19-24 2015, pp. 5833–5837.

[3] L. J. Rodriguez-Fuentes, A. Varona, M. Penagarikano, G. Bordel, and M. Diez, "High-performance query-by-example spoken term detection on the sws 2013 evaluation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2014)*, Florence, Italy, May 4-9 2014, pp. 7819–7823.

[4] P. Schwarz, "Phoneme recognition based on long temporal context," Ph.D. dissertation, Faculty of Information Technology, Brno University of Technology, http://www.fit.vutbr.cz/, Brno, Czech Republic, 2008.

[5] A. Abad, L. J. Rodriguez-Fuentes, M. Penagarikano, A. Varona, M. Diez, and G. Bordel, "On the calibration and fusion of heterogeneous spoken term detection systems," in *Interspeech 2013*, Lyon, France, August 25-29 2013, pp. 20–24.

[6] *Python interface to the WebRTC (https://webrtc.org/) Voice Activity Detector (VAD)*, https://github.com/wiseman/py-webrtcvad.

[7] A. Silnova, P. Matejka, O. Glembek, O. Plchot, O. Novotny, F. Grezl, P. Schwarz, L. Burget, and J. H. Cernocky, "BUT/Phonexia Bottleneck Feature Extractor," in *Odyssey 2018: The Speaker and Language Recognition Workshop*, Les Sables D'Olonne, France, June 26-29 2018, pp. 283–287.

[8] C. Cieri, D. Miller, and K. Walker, "The Fisher Corpus: a Resource for the Next Generations of Speech-to-Text," in *LREC*, 2004, pp. 69–71.

[9] *Babel Program*, Intelligence Advanced Research Projects Activity (IARPA), https://www.iarpa.gov/index.php/research-programs/babel.

[10] N. Brümmer and E. de Villiers, *The BOSARIS Toolkit User Guide: Theory, Algorithms and Code for Binary Classifier Score Processing*, 2011, https://sites.google.com/site/bosaristoolkit/.

[11] ——, "The BOSARIS Toolkit: Theory, Algorithms and Code for Surviving the New DCF," arXiv.org, 2013, presented at the NIST SRE'11 Analysis Workshop, Atlanta (USA), December 2011, https://arxiv.org/abs/1304.2865.

[12] *gTTS (Google Text-to-Speech): Python library and CLI tool to interface with Google Translate's text-to-speech API*, Download and installation: https://pypi.org/project/gTTS/. Documentation: https://gtts.readthedocs.io/en/latest/.

[13] *NSSpeechSynthesizer: The Cocoa interface to speech synthesis in macOS (AppKit module of PyObjC bridge)*, Apple Developer Objective C Documentation: https://developer.apple.com/documentation/appkit/nsspeechsynthesizer. Stackoverflow example of use (with Python 2): https://stackoverflow.com/questions/12758591/python-text-to-speech-in-macintosh.

[14] J. Tejedor and D. T. Toledano, *The ALBAYZIN 2018 Search on Speech Evaluation Plan*, IberSpeech 2018: X Jornadas en Tecnologías del Habla and V Iberian SLTech Workshop, Barcelona, Spain, November 21-23 2018, http://iberspeech2018.talp.cat/wp-content/uploads/2018/06/EvaluationPlanSearchonSpeech.pdf.

[15] *MAVIR corpus*, Laboratorio de Lingüística Informática, Universidad Autónoma de Madrid, http://www.lllf.uam.es/ESP/CorpusMavir.html.

[16] E. Lleida, A. Ortega, A. Miguel, V. Bazán, C. Pérez, M. Zotano, and A. de Prada, *RTVE2018 Database Description*, Vivolab, Aragon Institute for Engineering Research (I3A), University of Zaragoza and Corporación Radiotelevisión Española, June 2018, http://catedrartve.unizar.es/reto2018/RTVE2018DB.pdf.

[17] *COREMAH corpus*, Laboratorio de Lingüística Informática, Universidad Autónoma de Madrid, http://www.lllf.uam.es/coremah/.

[18] T. Hazen, W. Shen, and C. White, "Query-By-Example Spoken Term Detection Using Phonetic Posteriorgram Templates," in *ASRU*, Merano, Italy, December 13-17, 2009, pp. 421–426.