Epistemic Injustice as a Philosophical Conception for Considering Fairness and Diversity in Human-centered AI Principles

doi: 10.4036/iis.2022.A.01

Interdisciplinary Information Sciences Vol. 28, No. 1 (2022) 1–13

ISSN 1340-9050 print/1347-6157 online
DOI 10.4036/iis.2022.A.04

Spoken Term Detection of Zero-Resource Language Using Posteriorgram of Multiple Languages

Satoru MIZUOCHI[1], Takashi NOSE[1] and Akinori ITO[1,2,*]

[1]*Graduate School of Engineering, Tohoku University, Sendai 980-8579, Japan*
[2]*Advanced Institute for Yotta Informatics, Tohoku University, Sendai 980-8579, Japan*

We propose in this paper a query-by-example spoken term detection (QbE-STD) method for keyword detection from zero-resource language speech databases. The proposed method employs the phonetic posteriorgram (PPG) trained with multiple resource-rich languages and combines multilingual PPGs for speech representation. The keywords are detected using the dynamic time warping method. We examined three types of combination of multiple languages such as concatenation of PPG (PPG_CONC), a combination of language resources to calculate multilingual PPG (PPG_ALL), and multi-task training of PPG using multiple languages (PPG_DIV). We carried out an experiment of the QbE-STD from Kaqchikel speech. As a result, the use of PPG showed better detection performance than the method based on the conventional speech feature (MFCC), and the use of multiple languages gave a further improvement of detection.

KEYWORDS: spoken term detection, query by example, zero-resource language, Kaqchikel, phonetic posteriorgram

1. Introduction

Many minority languages are currently in danger of extinction. About 4,000 to 6,000 languages are said to exist globally, and many are expected to become extinct within this century [19]. Although it is difficult to stop a language from disappearing, people have made many attempts to document it while their speakers are still around [14, 15, 20, 38, 39]. In addition to textual languages, it is important to preserve spoken languages, and efforts to preserve many languages are underway. Some of them are ambitious, such as the Human Language Project [1], which archives all the world's languages, while others use the developed corpora of the endangered languages [24].

It is desirable to record and catalog speech data in such databases and search the database by words. There are several approaches to search words in speech databases. One approach is transcribing the speech database to be searched by a speech recognizer and then the text search [7]. This method enables a quick search using text. This method uses an accurate speech recognizer of the target language, and the development of such a recognizer requires a large amount of language resources.

However, most minority languages do not have such a large corpus used for speech recognizer development. Such languages are called "low-resource languages." Although several studies have developed speech recognition systems for low-resource languages [5], it is generally difficult to realize a high-precision speech recognizer without good language resources (recorded in a clean environment and transcribed). Moreover, the endangered languages often have no such resource at all. When language resource for developing a speech recognizer is not available, such language is called "a zero-resource language [23]." We are not able to use a speech recognizer for such language. Thus, several methods have been developed that search the speech database without transcribing the speech in the database. These methods use speech input as the search term and search the database using signal processing methods [3, 4, 25]. The method of searching for spoken keywords, called "Query-by-Example Spoken Term Detection (QbE-STD)," is recognizer-free and can be realized regardless of the availability of language resources of the target language.

These technologies, which organize unstructured data containing unknown contents, are essential for developing new value from the data. Although we assume that we cannot use any resource of a zero-resource language to develop a speech recognizer, it does not necessarily mean that the amount of speech data is small. For example, if we record all the speakers' utterances of the language, the amount of the recorded data becomes unmanageable [26]. Therefore, we need to develop methods to evaluate huge data of unknown contents [34]. A similar idea can be useful for image data, not only speech data [30].

In this paper, we propose a language-independent QbE-STD method for zero-resource languages. An QbE-STD method for an unknown language should have two properties: speaker-independent and language-independent

Received September 29, 2021; Accepted March 11, 2022
Part of this work was supported by JSPS KAKENHI Grant Number JP19H05589.
*Corresponding author. E-mail: aito@spcom.ecei.tohoku.ac.jp

detection. Conventional methods extract speaker-independent features from the speech signal, such as a vector of posterior probabilities of phonemes (posteriorgram [11]), to remove the speaker variation from the feature. However, posteriorgram is language-dependent. On the other hand, the signal-processing-based features, such as Mel-frequency Cepstral Coefficients (MFCC), are independent of language, but they are affected by speakers. Thus, we combine multiple posteriorgrams for multiple languages to obtain a feature independent of both speakers and languages.

First, the proposed method extracts posteriorgrams trained using two languages, English and Japanese, for both speech signals in the database and the query speech. Then we apply a query speech detection algorithm based on the continuous DP matching that calculates the similarity between the query speech and the segments in the database speech. Finally, the parts of the database speech similar to the query speech are detected as the search result.

We assume Kaqchikel [6, 18] as the target language in this research. Kaqchikel is the Mayan language spoken in Guatemala, with 450,000 speakers.

2. QbE-STD Using Nonlinear Matching

2.1 Overview

As described above, the proposed method uses the dynamic time warping for continuous speech, or "continuous dynamic programming (CDP)," as the basic algorithm for detecting query speech from the database. Therefore, we explain how a spoken keyword can be detected from a speech database. Detection of words from continuous speech originated in the 1980s. At that time, this task was called "word spotting." The nonlinear matching, such as Dynamic Time Warping (DTW) between the spoken keyword and the speech in the database, was executed to perform the word spotting [27]. In this method, the speech database and keyword speech are converted into spectrum-based feature vector sequences such as LPC mel-cepstra or MFCC, and the DTW finds the correspondence that minimizes the accumulated distance between feature vectors of the keyword speech and the database speech.

2.2 Detection of words using DTW

Dynamic Time Warping (DTW) is a method to find matches between the spoken input and the speech in the database. The basic DTW matches the feature vectors of the input speech $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_I$ and the spoken keyword input $\mathbf{Y} = \mathbf{y}_1, \ldots, \mathbf{y}_J$ as follows. First, let us consider the correspondence Φ, which is a sequence of pairs of indices in $\mathbf{X}$ and $\mathbf{Y}$:

$$\Phi = (\phi_1, \ldots, \phi_N) \tag{1}$$

$$\phi_k = (i_k, j_k),\ 1 \le i_k \le I,\ 1 \le j_k \le J \tag{2}$$

$$\phi_1 = (1, 1),\ \phi_N = (I, J) \tag{3}$$

Here, $\phi_k = (i_k, j_k)$ is a pair of corresponding indices, i.e., $\mathbf{x}_{i_k}$ corresponds to $\mathbf{y}_{j_k}$. The indices i_k and j_k increase monotonically with respect to k, i.e., $i_k \le i_{k+1}$ and $j_k \le j_{k+1}$. Then the DTW finds the correspondence with minimum distance, as follows. Let $D(\mathbf{X}, \mathbf{Y}, \Phi)$ be the accumulated distance between $\mathbf{X}$ and $\mathbf{Y}$ under the correspondence Φ:

$$D(\mathbf{X}, \mathbf{Y}, \Phi) = \sum_{k=1}^{N} d(\mathbf{x}_{i_k}, \mathbf{y}_{j_k}) \tag{4}$$

Here, $d(\mathbf{x}, \mathbf{y})$ is the Euclidean distance between two vectors $\mathbf{x}$ and $\mathbf{y}$. Then we want to find the correspondence $\widehat{\Phi}$ that minimizes D.

$$\widehat{\Phi} = \arg\min_{\Phi} D(\mathbf{X}, \mathbf{Y}, \Phi) \tag{5}$$

This optimization problem can be solved using dynamic programming, as follows [33].

$$d_{i,j} = \begin{cases} d(\mathbf{x}_i, \mathbf{y}_j) & 1 \le i \le I \text{ and } 1 \le j \le J \\ \infty & \text{otherwise} \end{cases} \tag{6}$$

$$g(i, j) = -\infty \quad \text{for } i \le 0 \text{ or } j \le 0 \tag{7}$$

$$g(1, 1) = d_{1,1} \tag{8}$$

$$g(i, j) = d_{i,j} + \min \begin{cases} g(i-2, j-1) + d_{i-1,j} \\ g(i-1, j-1) \\ g(i-1, j-2) + d_{i,j-1} \end{cases} \tag{9}$$

$$D(\mathbf{X}, \mathbf{Y}, \widehat{\Phi}) = g(I, J) \tag{10}$$

Here, $d_{i,j}$ is the distance matrix between the database speech $\mathbf{X}$ and query speech $\mathbf{Y}$, and $g(i, j)$ is the accumulated distance matrix in which $g(i, j)$ is the smallest accumulated distance at (i, j). Then $\widehat{\Phi}$ is determined as the best matching path obtained by tracing the minimum decisions in Eq. (9).

To detect a keyword from a longer speech database, we need to apply a different approach from the DTW mentioned above since the original DTW matches the vector sequences with almost similar lengths. This algorithm is called "the

continuous DP matching" [27, 29].

Let us assume that $I \gg J$, and

$$\mathbf{X}(i', i) \equiv (\mathbf{x}_{i'}, \ldots, \mathbf{x}_i) \tag{11}$$

Then the detection score of $\mathbf{Y}$ from $\mathbf{X}$ at position i is obtained as follows.

$$D(\mathbf{X}, \mathbf{Y}, i) = \min_{i'} \min_{\Phi} D(\mathbf{X}(i', i), \mathbf{Y}, \Phi) \tag{12}$$

This score can be calculated in almost the same way as Eq. (7). The difference is that the start and endpoints of sequence $\mathbf{Y}$ are not fixed to the start and endpoints of $\mathbf{X}$. The score can be obtained by

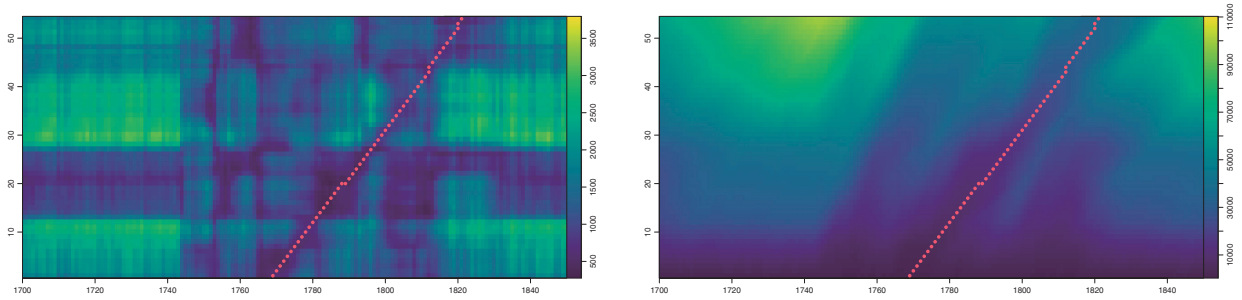
$$D(\mathbf{X}, \mathbf{Y}, i) = g(i, J). \tag{13}$$

Figure 1 shows examples of the distance matrix d, accumulated distance matrix g, and the optimum correspondence (the DP path) Φ shown as the red dots. In these figures, the dark color shows a short distance. Figure 1(a) shows that it is clear that the DP path travels along the "valley" of the distance matrix.

After calculating the score $D(\mathbf{X}, \mathbf{Y}, i)$, we find the local minima of the score and determine the minima with the scores lower than a threshold are detected as the candidate of the terms.

Figure 2 shows an example of term detection to explain how the keywords are detected and how the speaker variability affects the performance of QbE-STD. In this example, the spoken word "*matyöx*" (thank you) is detected from the continuous speech in the database. Two speakers speak the same keyword: one is the same speaker as the speech in the database, and the others are a different speaker. The red and green curves show the scores for the keyword of the different speaker and the same speaker, respectively. The blue lines are the last points of the keywords in the database. In this figure, we can see that the minima of the different speaker's scores are larger than that of the same speaker, and it is difficult to detect only the keyword positions when the speakers of the keyword and the database are different.

As reviewed in the introduction, information of the phoneme and the speaker is involved in a spectral feature vector. Therefore, if the speakers of the database and the keyword are different, the distance between the feature vectors reflects not only the difference of pronunciation but also speaker variation of the speech. The difference caused by speaker variation is often larger than that from pronunciation, which is the main cause of the keyword detection performance degradation. Ito and Koizumi proposed a method that uses a neural network to calculate the speaker-



(a) The distance matrix d (b) The accumulated distance matrix g

Fig. 1. The distance matrix, accumulated distance matrix and the optimum DP path.

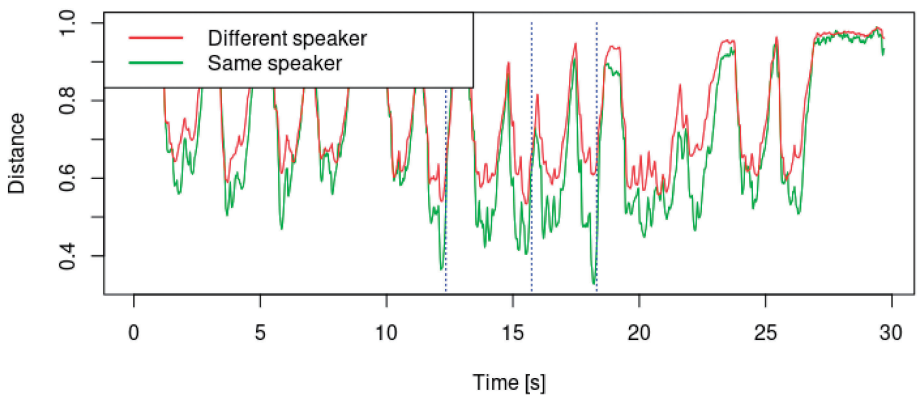


Fig. 2. An example of term detection using DTW. The red and green lines are scores for the keywords, and the blue lines are the end positions of the keyword.

independent distance between the MFCC features [12], but this method is computationally expensive, and the performance is insufficient.

3. The Keyword Detection Based on the Phonetic Posteriorgram

3.1 The posteriorgram

The problem with the QbE-STD based on spectral features was that the distance between the speech features depends on pronunciation difference and speaker variation. Therefore, in order to reduce the influence of differences of feature vectors induced by the speaker difference, the feature vectors are converted into the speaker-independent features, and then the distance between the speaker-independent features is calculated.

The posteriorgram [11, 31, 37] is a speaker-independent feature representation of short-time speech. Let us assume that a sequence of short-time (around 10 to 20 ms) speech signals (or spectral feature extracted from the signal, such as MFCC) is given as $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$. It can be classified into a phonological class $\omega \in \Omega = \{\omega_1, \ldots, \omega_K\}$. The class Ω can be a set of target language phonemes [11], articulatory features [31], or states of hidden Markov models (HMM) for phoneme recognition [28]. Then a posteriorgram is a vector of posterior probabilities:

$$\boldsymbol{p}_t = (P(\omega_1|\mathbf{x}_t), \ldots, P(\omega_K|\mathbf{x}_t)) \tag{14}$$

where

$$\sum_{k=1}^{K} P(\omega_k|\boldsymbol{x}) = 1. \tag{15}$$

We use the sequence of posteriorgrams $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_T$ as the feature vector sequence.

When we choose phonemes as the classes, the feature is called the phonetic posteriorgram (PPG). The probabilities are calculated using a classifier, such as the Gaussian mixture model (GMM) [32] or the deep neural network (DNN) [28]. Since the classification accuracy of the DNN generally outperforms that of the GMM, it is reasonable to choose DNN as the classifier.

If we train the classifier using the speech of multiple speakers, the classifier is expected to ignore the speaker variation, and the posteriorgram contains only linguistic variation.

The use of posteriorgram can realize the speaker-invariant feature vectors that express only phonetic variation. However, since the posteriorgram uses phonetic information, it inevitably depends on the target language. A few attempts have been developed to develop a multilingual QbE-STD using posteriorgram [7], such as articulatory features [31], but it only works for languages that share the articulatory features. The difficulty of Kaqchikel is that it has glottal plosive, and most of the major languages do not have that feature. A similar idea was proposed by Tanaka *et al.* [36] that converts the speech into a language-independent phonetic symbol sequence, but it has a similar problem with the articulatory-feature-based posteriorgram.

This section examines how the PPG for different languages works for QbE-STD. We used Japanese and English PPGs for detecting words of Kaqchikel.

Figure 3 shows how we extract the PPGs from the input and database speech signals. First, we train phoneme classifiers for languages with a large amount of training data. In this work, we use English and Japanese. Then the input query and the database speech are classified frame by frame, and the outputs of the classifier, the PPGs, are used as the feature of QbE-STD. Finally, we use MFCC and its first derivative as the input to the classifier.

3.2 Training data and model structure

We used the Japanese Newspaper Article Sentences (JNAS) [13] as the training data of the Japanese phoneme classifier. We chose the phone-balanced 2794 sentences uttered by 127 male and 127 female speakers as the training data (4.17 h), 286 sentences uttered by 13 male and 13 female speakers as the development data, and 286 sentences uttered by 13 male and 13 female speakers as the evaluation data.

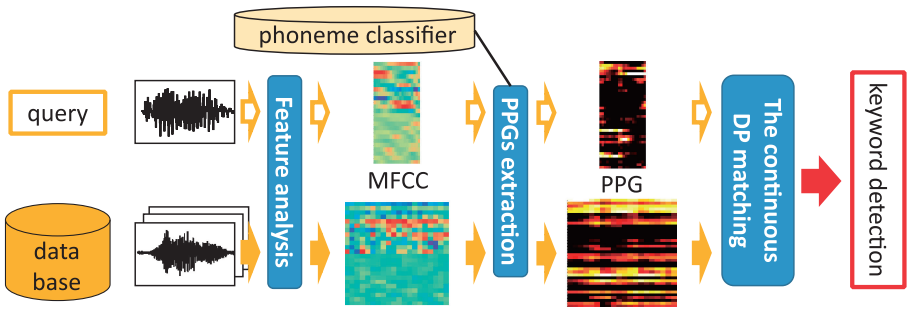


Fig. 3. Block diagram of the PPG extraction system.

TIMIT [9] was used to train the English phoneme classifier. TIMIT is a phonetic corpus of native speakers of American English who differ in gender and dialect. We used 4900 sentences uttered by 344 male and 146 female speakers as training data (4.19 h), 400 sentences uttered by 27 male and 13 female speakers as development data, and 400 sentences uttered by 27 male and 13 female speakers as evaluation data.

On training the model, we balanced the number of samples for phonemes by repeating the small-numbered samples.

A Multilayer Perceptron (MLP) was used as the phoneme classifier. The outputs are the classes of phonemes, and the input is a segment of the Mel-frequency cepstral coefficients (MFCC) with its first derivatives. The MFCCs were extracted with a sampling frequency of 16 kHz, a window length of 25 msec, and a frameshift of 10 ms. In addition, temporal derivatives of MFCCs (ΔMFCC) were used, too. In total, the dimension of the feature was 24. Contiguous feature vectors are given as an input of the network. Let the feature vectors of the speech be $\mathbf{x}_1, \ldots, \mathbf{x}_T$. When the segment width is $2k+1$, we used the segment $\mathbf{X}_i = \mathbf{x}_{i-k}, \ldots, \mathbf{x}_{i+k}$ as the input vector. The common conditions for all languages are shown in Table 1(a). We developed four phoneme classifiers corresponding to four input frame widths (1, 9, 15, and 17) for Japanese and English. The number of hidden layers and the number of nodes in the hidden layers were optimized using Optuna [2]. The determined hyperparameters are listed in Table 1(b) and (c).

Figure 4 shows the frame-by-frame phoneme recognition rate. The recognition rate of the English classifier was lower than that of the Japanese one. There could be multiple reasons of the difference of phoneme recognition rate. Since the amount of training data is comparable, one possible reason is that English has more phonemes than Japanese. This result is consistent with other multilingual phone recognition results [21, 35].

3.3 Conditions for word detection experiment

We compared the performance of word detection when using MFCC and Japanese/English PPGs.

Table 1. Hyperparameter setting and determined hyperparameters for PPG extraction models.

(a) Common conditions for all languages

Input frames	1, 9, 15, 17
Number of hidden layers	2 to 7
Number of hidden units	512, 1024, 2048, 4096, 8192
Optimizer	Adam
Activation function	ReLU
Dropout probability	0.5

(b) Determined conditions for Japanese

Input frames	1	9	15	17
Number of hidden layers	4	4	3	5
Number of hidden units	1024			
Phonetic classes	36			

(c) Determined conditions for English

Input frames	1	9	15	17
Number of hidden layers	4	3	3	4
Number of hidden units	1024	2048	1024	1024
Phonetic classes	46			

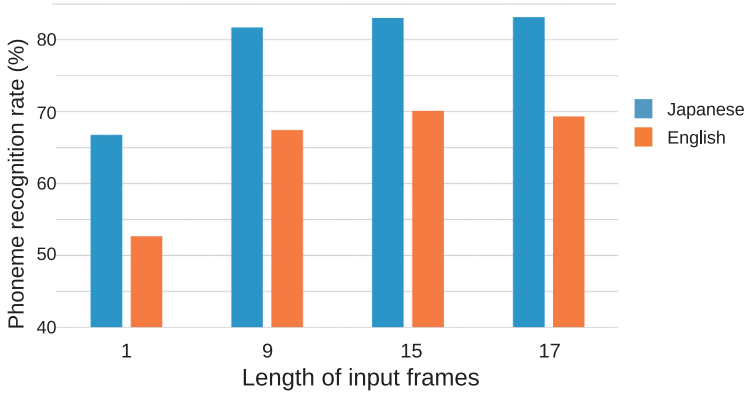


Fig. 4. The frame-by-frame phoneme recognition rate for Japanese and English speech using different frame widths.

In the experiments, we first checked the performance of QbE-STD when the target language was Japanese or English and then evaluated the performance when the target language was Kaqchikel. We compared the following three features: 12-dimension MFCC and their first derivatives, which is the same feature as PPG calculation (denoted as MFCC), Japanese PPG (denoted as PPG_JP), and English PPG (denoted as PPG_EN).

For evaluation of Japanese, we used six utterances as queries, "*azia*" (Asia), "*kankei*" (relation), "*senkyo*" (election), "*tyoosa*" (survey), "*nihon*" (Japan), and "*paasento*" (percentage), uttered by speakers not included in the evaluation database. For evaluation of English, we used six utterances as queries, "dark," "greasy," "suit," "wash," "water," and "year" uttered by speakers not included the evaluation database. These queries were selected so that the words appear in the database frequently and are not too short. For the evaluation data of the Kaqchikel language, we used 347 utterances of 16 dialogues of Kaqchikel language recorded in a quiet environment (about 14 min). The query utterances were "*achike*," "*matyöx*," "*peraj*," and "*richin*," and these speakers are not included in the evaluation database.

These query utterances were spliced out of continuous speech. Therefore, we do not have silence before and after the query speech. When the queries were converted into feature sequences, we padded the silence before and after the query speech with white noise with small amplitude so that we could take the segment from the beginning to the end of the original query speech. For example, when we have $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_T$ and the segment length is 7, we added a small white noise signal so that $\mathbf{x}_{-2}, \mathbf{x}_{-1}, \mathbf{x}_0$ and $\mathbf{x}_{T+1}, \mathbf{x}_{T+2}, \mathbf{x}_{T+3}$ can be used.

We decided the correctness of the keyword detection on a per-utterance basis; in other words, we regarded that the keyword detection was correct if the detected keyword was included in the utterance, regardless of the position of detection. Thus the evaluation could be somewhat incorrect, but we think this criterion is not too incorrect because the utterances in the database are only several seconds long.

We used the Mean Average Precision (MAP) [8, 10, 17] as the evaluation index. MAP is the average of the Average Precision (AP) calculated for each query over all queries, and the larger the value, the more correct answers are at the top of the detection results. Assume that we have a query q, and we detect N occurrences of the keyword in the database, denoted as $o_1(q), \ldots, o_N(q)$. We assume that each occurrence has relevance score $S(o_i(q))$ (large value means strong relevance), and arranged by the score:

$$S(o_1(q)) \geq \cdots \geq S(o_N(q)). \tag{16}$$

Then we use the "true" relevance of o_i as $\delta(o_i(q))$, such that

$$\delta(o_i(q)) = \begin{cases} 1 & \text{if } o_i(q) \text{ is correct} \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

In addition, let c_q be the number of utterances to be detected for query q. AP and MAP are represented by the Eqs. (18) and (20), respectively.

$$AP(q) = \frac{1}{c_q} \sum_{i=1}^{N} \delta(o_i(q)) R_{prec}(q, i) \tag{18}$$

$$R_{prec}(q, i) = \frac{1}{i} \sum_{k=1}^{i} \delta(o_k(q)) \tag{19}$$

$$MAP = \frac{1}{Q} \sum_{q=1}^{Q} AP(q) \tag{20}$$

where c_q is the number of detected utterances containing the query q, $R_{prec}(q, i)$ is the precision when utterance i is detected in query q, and Q is the total number of queries.

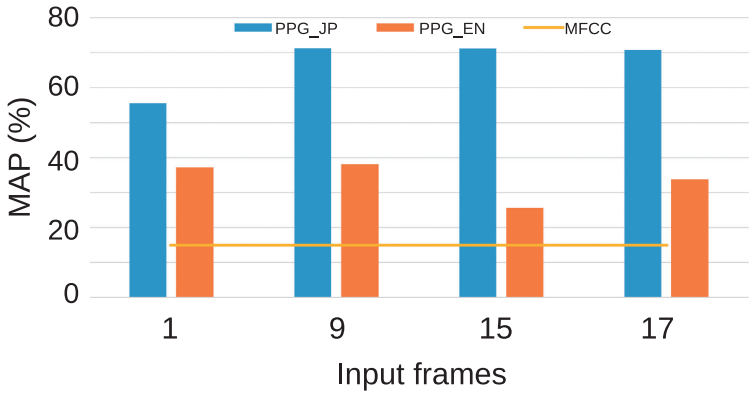
3.4 Results of the word detection experiment

The results of the word detection experiments for the target languages (Japanese and English) are shown in Figs. 5(a) and 5(b), respectively. The results show that the PPG improves the word detection accuracy compared to the MFCC in all conditions.

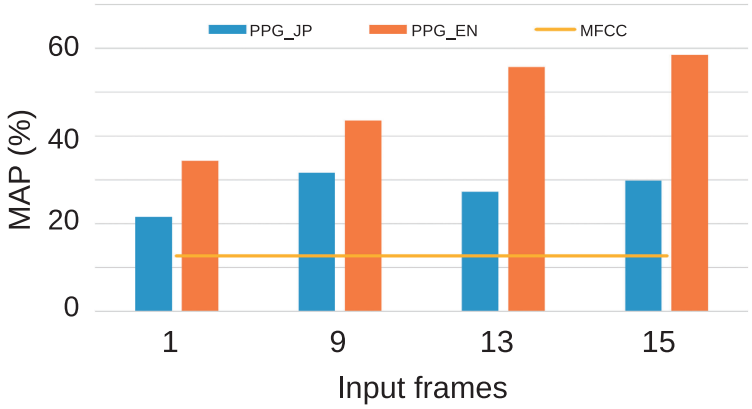
Next, we show the results of the word detection experiment for Kaqchikel in Fig. 5(c). Again, it is confirmed that the PPG improves the word detection accuracy more than the MFCC in almost all conditions.

As for the optimum number of frames, Figs. 5(a) and 5(b) suggest that 15 or 17 frames are sufficient when the training and detection languages are the same, which is consistent with the result shown in Fig. 4. For English, longer frame than 17 frames might give better result. However, in the language-mismatch condition (PPG_EN for Japanese utterances and PPG_JP for English utterances), increasing the number of frames does not necessarily increase the performance. We also can see this tendency in Fig. 5(c).

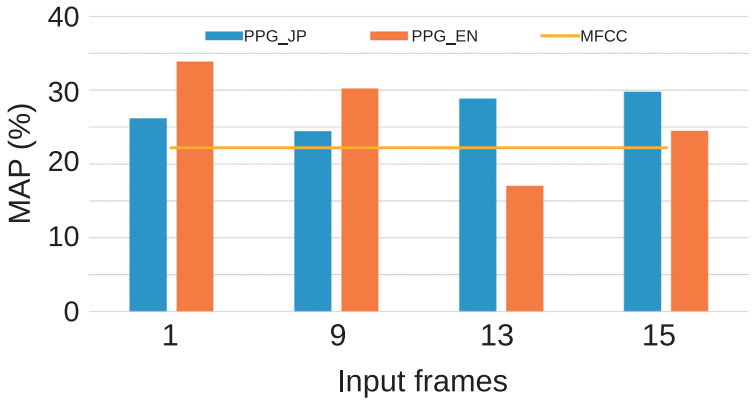
Figure 6 is a comparison of the best detection results for all languages and PPGs. This result confirms that improvement is smaller when the target language and the language used to train the model are different, suggesting that the linguistic difference between the target language and PPG significantly impacts QbE-STD performance. The reason



(a) Detection results for Japanese utterances



(b) Detection results for English utterances



(c) Detection results for Kaqchikel utterances

Fig. 5. Detection results (MAP) for Japanese, English and Kaqchikel utterances with different frame widths.

for this deterioration is not clear, but possible reasons are insufficient phoneme inventory (for example, Japanese phoneme inventory lacks some phonemes that are needed to describe English) or difference of phoneme boundary of two languages in the feature space.

4. Combination of Posteriorgrams of Multiple Languages

4.1 Overview

From the experimental results in the previous section, we confirmed that the use of PPG improved the QbE-STD performance and the language difference deteriorated the performance. Thus, to improve the QbE-STD performance without using the training data of the target language, we examine using PPG of multiple languages, expecting that the use of multiple languages compensates for the difference of phoneme inventory between the target and training languages.

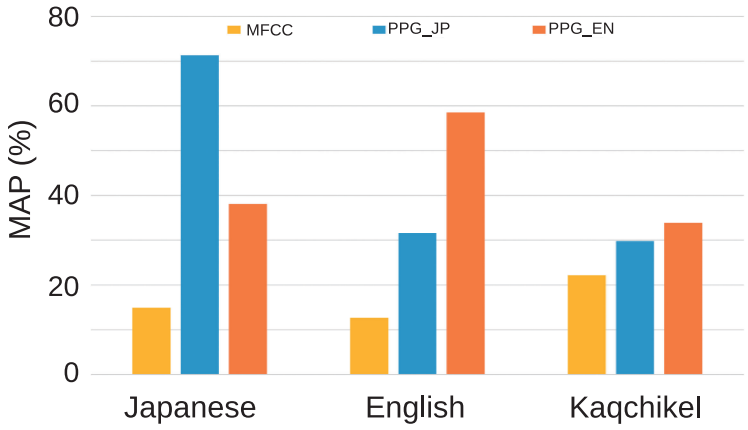


Fig. 6. MAP by detection experiments for three languages with optimum frame width.

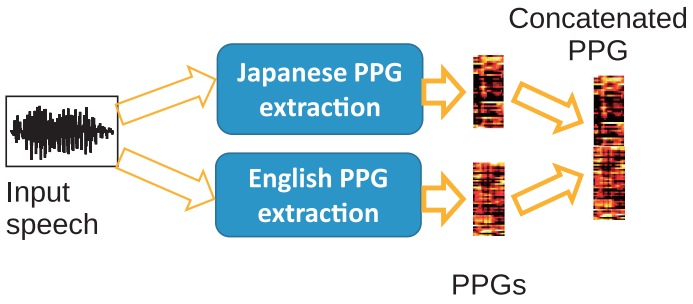


Fig. 7. Calculation of two-language PPG with the concatenation-based method.

We examined two methods for combining multiple languages. The first one is the concatenation-based method, where we train two phoneme classifiers independently and concatenate the output of the classifiers to obtain multilingual PPGs. The second one is the multilingual method, where we train a single model using training data of the two languages.

4.2 Calculation methods of multilingual PPG

Figure 7 shows an overview of the concatenation-based method. Here, we use English and Japanese PPGs to perform QbE-STD for Kaqchikel. First, we convert the speech in the database and the voice query into the PPGs using the phoneme classifiers of the training languages. Then we concatenate the PPGs extracted from the phoneme classifiers frame by frame. Therefore, one frame of the concatenated PPG has posterior probabilities of two languages, as follows.

$$\boldsymbol{p}_t = (P_{JPN}(\omega_1|\mathbf{x}_t), \ldots, P_{JPN}(\omega_K|\mathbf{x}_t), P_{ENG}(\omega_1|\mathbf{x}_t), \ldots, P_{ENG}(\omega_{K'}|\mathbf{x}_t)) \tag{21}$$

Here, $P_{JPN}(\omega_i|\mathbf{x})$ and $P_{ENG}(\omega_j|\mathbf{x})$ are the posterior probabilities of Japanese phoneme ω_i and English phoneme ω_j, respectively. Since these probabilities are independent, the sum of all values in this PPG becomes two.

Figure 8(a) shows PPG extraction using the multilingual method. In this method, we train the classifier to output multilingual PPG directly. Here, we examined two training methods. First, the "ALL" method regards the phonemes of different languages as different classes. Training of the ALL method is shown in Fig. 8(b). Here, we prepare the one-hot vectors (a vector where the value of only one dimension is one, and all the others are zero) for two languages as reference. For example, English /a/ and Japanese /a/ are trained as different phonemes in this method.

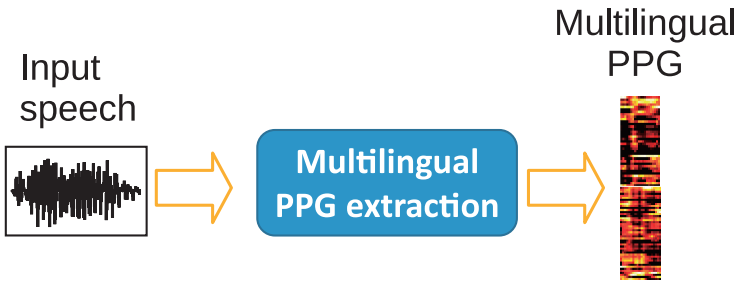
The second method, which we call the "DIV" method, trains phonemes of two languages independently. The training of the DIV method is shown in Fig. 8(c). When training a phoneme of one language, we prepare a one-hot vector for only that language as a reference, and the connections from the hidden layer to the output of another language are not trained. Thus, the "DIV" method is similar to the concatenated-based method; the difference is that the "DIV" method shares the hidden layers by the two languages.

4.3 Experimental conditions

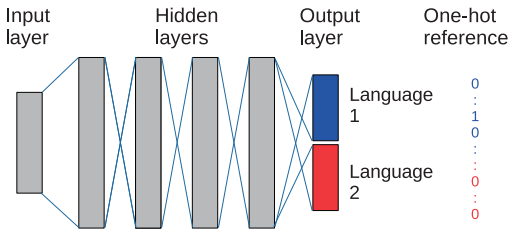
The training data used in the experiment were the same as that used in the previous section.

We used a fully connected multi-layer perceptron (MLP) as a classifier. Table 2 shows the parameters of the network. The output units correspond to 36 and 46 Japanese and English phoneme classes, respectively.

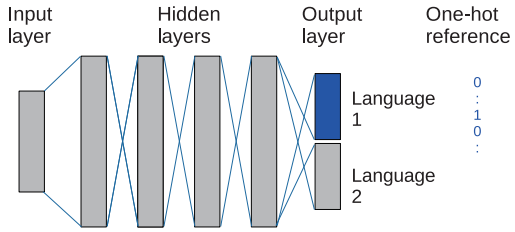
The feature vectors for PPG extraction was the 12-dimensional MFCC and their temporal derivatives, extracted under the conditions of a sampling frequency of 16 kHz, a window length of 25 msec, and a frameshift of 10 msec.



(a) Extraction of PPG by the multilingual method



(b) Training method (ALL)



(c) Training method (DIV)

Fig. 8. Calculation of PPG with the multilingual method (PPG_ALL and PPG_DIV).

Table 2. Hyperparameter settings for training.

Language	Japanese, English
Length of input frames	1, 9, 15, 17
Number of hidden layers	3, 4, 5, 6
Number of hidden units	512, 1024, 2048, 4096, 8192
Activation function	ReLU
Optimizer	Adam
Dropout probability	0.5

Table 3. The optimum values of the hyperparameters.

Language	PPG_ALL				PPG_DIV			
Length of input frames	1	9	15	17	1	9	15	17
Number of hidden layers	3	3	3	3	3	3	3	3
Number of hidden units	1024	1024	1024	1024	2048	1024	1024	2048

The PPG extractors used for the PPG_CONC method were the same as in the previous chapter. We optimized hyperparameters of the extractors for PPG_ALL and PPG_DIV. Table 2 shows the examined hyperparameters and other conditions. The hyperparameters were decided by using Optuna [2]. Table 3 shows the optimum numbers of the hidden layers and the hidden units.

4.4 Results

Figure 9 shows the best MAP for each evaluation data. In the figure, PPG_JP and PPG_EN are the results using Japanese and English single-language PPGs, PPG_CONC is the result for the concatenation-based method, and PPG_ALL and PPG_DIV are the multilingual methods with “ALL” and “DIV” training.

From the Japanese and English results, it is found that the concatenation-based method gave a comparable performance to the single-language results with the matching (Japanese–Japanese and English–English) conditions. Among the multilingual methods, “DIV” training was almost as good as the concatenation-based method, but the “ALL” method was worse, especially detection of Japanese keywords. One possible reason for the deterioration in PPG_ALL is that this model tries to discriminate Japanese and English phonemes even if these phonemes are similar. For example, PPG_ALL is trained under the assumption that Japanese /s/ and English /s/ are different, but these pronunciations are similar [16]. Therefore, when we observe a pronunciation that exists in both English and Japanese, it could be classified into both English and Japanese phonemes, making the extracted PPG unstable; in other words, the

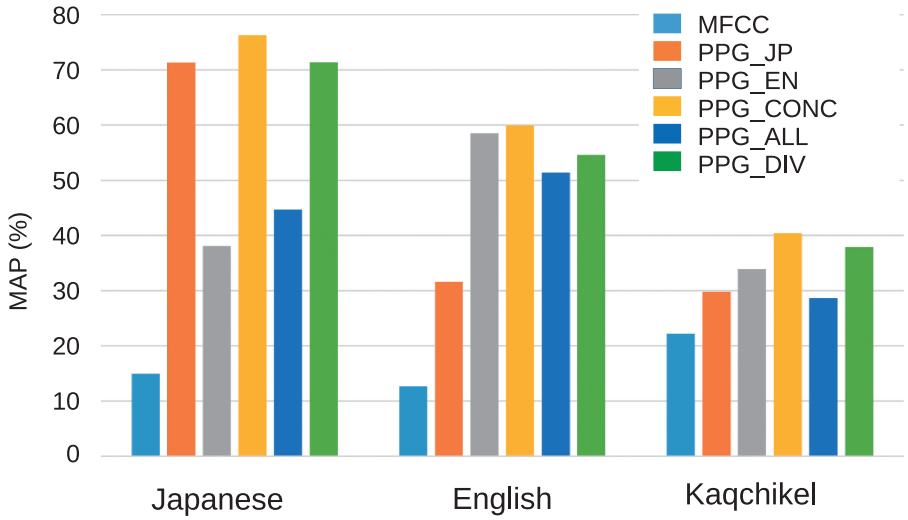


Fig. 9. Comparison of MAP with examined methods for three languages.

Table 4. Dimension of a feature vector.

Feature	D
MFCC	24
PPG_JP	36
PPG_EN	46
PPG_CONC	82

PPG_ALL model can output different PPG for similar pronunciation. Comparing PPG_CONC and PPG_DIV, PPG_CONC was slightly better.

The result for Kaqchikel is consistent with that for Japanese and English. PPG_CONC and PPG_DIV improved the performance compared to single-language models, and PPG_CONC was slightly better than PPG_DIV. We confirm that the detection performance for Kaqchikel is comparable to the performance when using a single-language model different from the target language (such as PPG_EN for Japanese), showing that the combination of Japanese and English is still insufficient to express phonetic variation of Kaqchikel.

4.5 On computation time and memory requirement

In the previous section, we observed the detection performance of the conventional and proposed methods. Other aspects of evaluating QbE-STD methods are computational complexity (calculation speed) and memory consumption [22]. In our work, we mainly focus on increasing the detection performance since the detection performance of QbE-STD for zero-resource language is not sufficient. This section overviews the computational complexity, calculation speed and memory complexity.

The conventional and proposed methods have two stages for detection: database preparation and query execution. The database preparation is usually performed only once before setting up the QbE-STD system, so we do not focus on this stage. The execution stage has two processes, feature extraction (MFCC and posteriorgram) and detection using continuous DP matching. Most of the calculation time is for the latter process, so we focus on the calculation of the continuous DP matching.

Let I and J be the length of the feature vector sequence of the database and the query, respectively, and suppose we have N_d data in the database. Let D be the dimension of the feature vector. Then the computational complexity of the continuous DP matching is $O(N_dIJD)$. Since the difference between the conventional and proposed methods is the feature vector, the calculation time depends on the dimension of the feature vector D. Table 4 shows the dimension of feature vectors of the four methods: MFCC, PPG_JP, PPG_EN, and PPG_CONC. As shown, the PPG_CONC method has a longer feature vector than the conventional method (MFCC); thus it is expected that PPG_CONC is almost 3.4 times slower than the MFCC-based method.

To investigate the actual impact of the feature dimension on the calculation time, we measured the actual calculation time of the QbE-STD. The experiment was conducted on a machine with Xeon E5-1650 v4 3.60GHz CPU, Ubuntu 16.04.7 LTS. The program was written in R 3.4.4 with tinyDP package*. The database was 16 utterances in Kaqchikel,

*https://github.com/akinori-ito/tinyDP

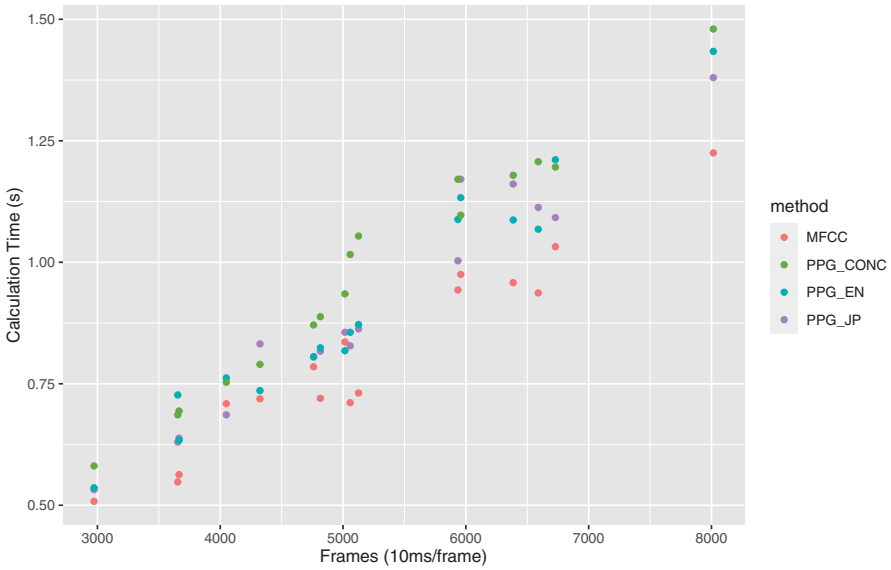


Fig. 10. Length of speech in the database and the calculation time.

Table 5. Analysis of calculation time.

	Estimate	Std. Error	t value	p-value
T_b	4.66×10^{-2}	6.80×10^{-2}	0.69	0.50
α	5.17×10^{-7}	2.45×10^{-7}	2.11	0.040
β	1.41×10^{-4}	1.27×10^{-5}	11.1	3.4×10^{-16}
γ	-1.44×10^{-4}	1.31×10^{-3}	-0.11	0.91

and we used the keyword "matyöx." The feature vectors were MFCC, PPG_JP, PPG_EN, and PPG_CONC. After measuring the time for DP matching, we analyzed the impact of dimension D and the length of database utterance I (in 10 ms frame). Figure 10 shows the calculation time with respect to the length of the speech signal in the database. As shown, the impact of dimension of the feature vector looks smaller as expected.

We conducted further analysis on the result. Although the computational complexity is $O(N_d IJD)$, the influence of I and D might be non-negligible for short sequence (i.e., the complexity is $O(ID + I + D)$). Therefore, we analyzed the calculation time T_d (s) using the least-squares method as

$$T_d = T_b + \alpha ID + \beta I + \gamma D \tag{22}$$

where T_b is the bias and α, β and γ are coefficients. The result is shown in Table 5. Here, p-value is the result of t-test whether the coefficient is nonzero or not. The coefficient γ and the bias T_b were not significant at 5% level, showing that the effect of feature dimension alone was small. Therefore, we can exclude the term with T_b and γ, and the calculation time can be approximated as

$$T_d = 5.17 \times 10^{-7} ID + 1.41 \times 10^{-4} I. \tag{23}$$

According to this approximation, for example, when the speech in database is 10 s long ($I = 1000$), the estimated calculation times for MFCC and PPG_CONC are 153 and 183 ms, respectively. Therefore, although the larger feature dimension increases the calculation time, its impact is limited.

This result implies that the calculation time of the recurrence formula [Eq. (7)] of the DP matching is comparable to that of distance matrix [Eq. (6)]. The computational complexity of the calculation of d by Eq. (6) is $O(IJD)$, while that of the recurrence formula Eq. (7) is $O(IJ)$. Thus, the total complexity becomes $O(IJ(D + const))$. If D were large, the constant would be negligible, and the computational complexity would be $O(IJD)$. The experimental result showed that the constant is not negligible for current matching algorithm implementation and the feature dimensions.

As for the memory requirement, the proposed method, especially PPG_CONC, requires more memory because its feature has larger dimension. However, the memory size of $d(i, j)$ and $g(i, j)$ in Eqs. (6) and (7) do not depend on the feature dimension because they only store the Euclidean distance.

5. Conclusion

We examined the QbE-STD method for zero-resource language using the PPGs of the multiple rich-resource languages. Here, we proposed a method that calculates PPGs of multiple languages. We examined three methods for calculating multilingual PPGs, and the concatenation-based method gave the best performance.

From the experimental result, we can draw the following conclusions.

When we use PPG for QbE-STD of Japanese and English, single-language PPG, in both language-match and language-mismatch conditions, improved the performance compared with MFCC. We confirmed that PPG can remove speaker variation from the feature from this result. However, in the language-mismatch conditions, the performance of QbE-STD deteriorates compared with the matching condition, possibly because of insufficient phoneme inventory (some phonemes are missing) or difference of phoneme boundary in the feature space. QbE-STD for Kaqchikel using the English or Japanese PPG was almost the same as the language-mismatch condition of English and Japanese.

When we used the multilingual PPG for Kaqchikel QbE-STD, the performance improved, which seems to be a result of compensating insufficient phoneme inventory. When combining PPGs of two languages, treating phonemes of different language as different units cause degradation of QbE-STD performance because treating very similar phonemes (such as Japanese /a/ and English /a/) as different units will make the classification unstable.

Finally, it can be said that the combination of English and Japanese is still insufficient to express Kaqchikel phonemes compared with the language-match condition of English and Japanese. Therefore, we need more languages to express the entire space of Kaqchikel phonemes.

We consider that the performance will improve when we use more languages for training the model. Therefore, in a future study, we will use more languages such as Chinese or Spanish to make further improvements.

Acknowledgment

Prof. Masatoshi Koizumi, Graduate School of Arts and Letters, Tohoku University, provided the Kaqchikel speech database. This study was supported by Yotta Informatics Project by MEXT, Japan, and JSPS Kakenhi JP19H05589.

REFERENCES

[1] Abney, S., and Bird, S., "The Human Language Project: Building a Universal Corpus of the World's Languages," *Proc. 48th Annual Meeting of the Association for Computational Linguistics*, 88–97 (2010).

[2] Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M., "Optuna: A Next-generation Hyperparameter Optimization Framework," *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2623–2631 (2019).

[3] Anguera, X., Rodriguez-Fuentes, L. J., Buzo, A., Metze, F., Szöke, I., and Penagarikano, M., "QUESST2014: Evaluating Query-by-Example Speech Search in a Zero-resource Setting with Real-life Queries," *Proc. Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 5833–5837 (2015).

[4] Barnard, E., Schalkwyk, J., van Heerden, C., and Moreno, P. J., "Voice Search for Development," *Proc. Interspeech* (2010).

[5] Besacier, L., Barnard, E., Karpov, A., and Schultz, T., "Automatic speech recognition for under-resourced languages: A survey," *Speech Communication*, **56**: 85–100 (2014).

[6] Brown, R. M., Maxwell, J. M., and Little, W. E., *La ütz awäch?: Introduction to Kaqchikel Maya Language*, University of Texas Press (2010).

[7] Deekshitha, G., and Mary, L., "Multilingual spoken term detection: A review," *International Journal of Speech Technology*, **23(3)**: 653–667 (2020).

[8] Garofolo, J. S., Auzanne, C. G., Voorhees, E. M., *et al.*, "The TREC spoken document retrieval track: A success story," *NIST SPECIAL PUBLICATION SP*, **500(246)**: 107–130 (2000).

[9] Garofolo, J. S., Lamel, L. F., Fisher, W. M., Fiscus, J. G., and Pallett, D. S., "DARPA TIMIT Acoustic-phonetic Continous Speech Corpus CD-ROM, NIST Speech Disc 1-1.1," *NASA STI/Recon Technical Report N 93* (1993) 27403.

[10] Hast, A., "Consensus Ranking for Increasing Mean Average Precision in Keyword Spotting," *VIPERC 2020, 2nd International Workshop on Visual Pattern Extraction and Recognition for Cultural Heritage Understanding*, Bari, Italy, 29 January, 2020, vol. 2602, 46–57 (2020).

[11] Hazen, T. J., Shen, W., and White, C., "Query-by-example Spoken Term Detection Using Phonetic Posteriorgram Templates," *Proc. IEEE Workshop on Automatic Speech Recognition & Understanding (ASRU)*, 421–426 (2009).

[12] Ito, A., and Koizumi, M., "Spoken Term Detection of Zero-resource Language Using Machine Learning," *Proceedings of the 2018 International Conference on Intelligent Information Technology*, 45–49 (2018).

[13] Itou, K., Yamamoto, M., Takeda, K., Takezawa, T., Matsuoka, T., Kobayashi, T., Shikano, K., and Itahashi, S., "JNAS: Japanese speech corpus for large vocabulary continuous speech recognition research," *Journal of the Acoustical Society of Japan (E)*, **20(3)**: 199–206 (1999).

[14] Janse, M., and Tol, S., Eds. Language Death and Language Maintainance: Theoretical, Practical and Descriptive Approaches, John Benjamins Publishing Co. (2003).

[15] Johnson, H., *Language Documentation and Description*, vol. 1. London: SOAS, 2003, ch. Language Documentation and Archiving, or How to Build a Better Corpus, 140–153.

[16] Kavanagh, B., "The phonemes of Japanese and English: A contrastive analysis study," *Aomori University of Health and*

Welfare, **8(2)**: 283–292 (2007).
[17] Kishida, K., "Property of average precision and its generalization: An examination of evaluation indicator for information retrieval experiments," *NII Technical Reports*, **2005(14)**: 1–19 (2005).
[18] Koizumi, M., Yasugi, Y., Tamaoka, K., Kiyama, S., Kim, J., Sian, J. E. A., and Mátzer, L. P. O. G., "On the (non) universality of the preference for subject-object word order in sentence comprehension: A sentence-processing study in Kaqchikel Maya," *Language*, **90(3)**: 722–736 (2014).
[19] Krauss, M., "The world's languages in crisis," *Language*, **68(1)**: 4–10 (1992).
[20] Laoire, S. N., "Scottish gaelic speech and writing: Register variation in an endangered language," *Scottish Language*, **27**: 115–119 (2008).
[21] Li, X., Dalmia, S., Li, J., Lee, M., Littell, P., Yao, J., Anastasopoulos, A., Mortensen, D. R., Neubig, G., Black, A. W., *et al.*, "Universal Phone Recognition with a Multilingual Allophone System," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 8249–8253 (2020).
[22] Mandel, A., Kumar, K. R. P., and Mitra, P., "Recent developments in spoken term detection: A survey," *Int. J. Speech Technol.*, **17(2)**: 183–198 (2014).
[23] Martin, L. J., Wilkinson, A., Miryala, S. S., Robison, V., and Black, A. W., "Utterance Classification in Speech-to-speech Translation for Zero-resource Languages in the Hospital Administration Domain," *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 303–309 (2015).
[24] McEnery, T., Baker, P., and Burnard, L., "Corpus Resources and Minority Language Engineering," *Proc. the Second International Conference on Language Resources and Evaluation (LREC'00)* (2000).
[25] Metze, F., Anguera, X., Barnard, E., Davel, M., and Gravier, G., "Language independent search in mediaeval's spoken web search task," *Computer Speech & Language*, **28(5)**: 1066–1082 (2014).
[26] Muraoka, H., Gyoba, G., Suzuki, Y., Shioiri, S., Nakao, M., Nihei, M., and Shoji, H., "Gigantic Amount Information and Storage Technology: Challenge to Yotta-byte-scale Informatics," *IEICE Technical Report*, vol. 116, 27–32 (2017).
[27] Nakagawa, S., "Connected spoken word recognition algorithms by constant time delay DP, O(n) DP and augmented continuous DP matching," *Information Sciences*, **33(1–2)**: 63–85 (1984).
[28] Obara, M., Kojima, K., Tanaka, K., Lee, S.-w., and Itoh, Y., "Rescoring by Combination of Posteriorgram Score and Subword-matching Score for Use in Query-by-Example," *INTERSPEECH*, 1918–1922 (2016).
[29] Oka, R., "Spotting method for classification of real world data," *The Computer Journal*, **41(8)**: 559–565 (1998).
[30] Oodaira, K., Miyazaki, T., Sugaya, Y., and Omachi, S., "Importance estimation for scene texts using visual features," *Interdisciplinary Information Sciences* (in press).
[31] Prabhavalkar, R., Livescu, K., Fosler-Lussier, E., and Keshet, J., "Discriminative Articulatory Models for Spoken Term Detection in Low-resource Conversational Settings," *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, 8287–8291 (2013).
[32] Reddy, P. R., Rout, K., and Murty, K. S. R., "Query Word Retrieval from Continuous Speech Using GMM Posteriorgrams," *2014 International Conference on Signal Processing and Communications (SPCOM)*, 1–6 (2014).
[33] Sakoe, H., and Chiba, S., "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **26(1)**: 43–49 (1978).
[34] Shioiri, S., Sato, Y., Horaguchi, Y., Muraoka, H., and Nihei, M., "Quali-informatics in the Society with Yotta Scale Data," *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 1–4 (2021).
[35] Siniscalchi, S. M., Svendsen, T., and Lee, C.-H., "Toward a Detector-based Universal Phone Recognizer," *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4261–4264 (2008).
[36] Tanaka, K., Itoh, Y., Kojima, H., and Fujimura, N., "Speech Data Retrieval System Constructed on a Universal Phonetic Code Domain," *IEEE Workshop on Automatic Speech Recognition and Understanding, 2001, ASRU'01*, 323–326 (2001).
[37] Wang, H., Lee, T., Leung, C. C., Ma, B., and Li, H., "Using Parallel Tokenizers with DTW Matrix Combination for Low-resource Spoken Term Detection," *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, 8545–8549 (2013).
[38] Woodbury, A. C., *Language Documentation and Description*, vol. 2. London: SOAS, 2003, ch. Defining documentary linguistics, 140–153.
[39] Yang, M.-C., and Rau, D. V., "An Integrated Framework for Archiving, Processing and Developing Learning Materials for an Endangered Aboriginal Language in Taiwan," *Proc. the Fifth Workshop on Asian Language Resources (ALR-05) and First Symposium on Asian Language Resources Network (ALRN)*, 32–39 (2005).