Simultaneous Flexible Keyword Detection and Text-dependent Speaker Recognition for Low-resource Devices

Hiroshi Fujimura, Ning Ding, Daichi Hayakawa and Takehiko Kagoshima
Toshiba Research and Development Center, Komukai-Toshiba-cho 1, Saiwai-ku, Kawasaki, 212-8582 Japan

Keywords: Keyword Detection, Speaker Identification, Low Resource Device, Bayesian.

Abstract: This paper proposes a new method for simultaneous flexible keyword detection and text-dependent speaker identification using a recognized keyword. The purpose is to identify a speaker from among a set of pre-registered speakers on the basis of a short-command utterance in an office or home on low-resource chip devices. The first contribution is to construct the process that includes a neural network (NN) and a customized Viterbi-based algorithm for flexible keyword detection, and Gaussian mixture models (GMMs) for speaker identification. Outputs of a middle layer in the NN and alignment information for keyword detection are also used for creating feature vectors for speaker GMMs. The second contribution is to apply DropConnect in speaker-modeling uncertainties of the Bayesian NN that is used for speaker reacognition. It results in robust speaker models when enrollment utterances are few. Evaluation was conducted using 39 Japanese keywords by 100 speakers. Recognition performance was measured on the basis of false acceptances and false rejects using keyword utterances. Speaker identification for 100 pre-registered speakers for recognized keywords was simultaneously evaluated. The identification rate when using a conventional i-vector method was 71.22%. By contrast, the identification rate of the proposed method was 89.29% while using low-cost resources.

1 INTRODUCTION

In recent years, there has been an increasing demand for keyword detection and speaker identification working together as part of a user interface for small devices (Wu et al., 2018; Chen et al., 2014; Variani et al., 2014; Chen et al., 2015b). They are being developed for smart speakers, home appliances, and other devices in the home or the office. Some are implemented on an embedded chip. The use of wake up words tends to be resistant to false acceptance errors. However, direct control of certain specific devices by voice command without the use of wake up words is more convenient. Moreover, many convenient applications can be created if a device identifies a speaker.

For example, a user only says “Turn on A/C” to start an air conditioner with a setting of comfortable temperature for the user. In this kind of scenario, the speaker identification has only to select one speaker from among a set of registered speakers using the recognized keyword utterance.

Many studies focus on keyword detection and speaker identification, independently. However, they should be simultaneously considered for implementing into low-resource devices in the real world, and it is important both from theoretical and application perspectives.

An i-vector representation is widely used for speaker identification (Dehak et al., 2011). However, an i-vector is generally designed to be used on speech lasting for at least 2 s (Kanagasundaram et al., 2011; Poddar et al., 2015; Tsujikawa et al., 2017). On the other hand, the duration of keywords for controlling devices is usually about less than 1 s. Thus, it is difficult to achieve good enough accuracy of speaker identification using keywords by an i-vector method.

Some methods for applying i-vectors to short keywords using a training data-set based on target keywords have been proposed (Variani et al., 2014; Larcher et al., 2012). Another approach for speaker identification using a short keyword is to introduce neural network (NN)-based features (Variani et al., 2014; Chen et al., 2015b; Chen et al., 2015a; Snyder et al., 2018). In (Variani et al., 2014; Chen et al., 2015b), a NN for speaker identification is first constructed for a specific keyword using substantial data on the keyword. Then, the NN outputs probabilities for all speaker IDs. During enrollment and evaluation, the method calculates output activations of the last hidden layer that includes speaker information for

Fujimura, H., Ding, N., Hayakawa, D. and Kagoshima, T.
Simultaneous Flexible Keyword Detection and Text-dependent Speaker Recognition for Low-resource Devices.
DOI: 10.5220/0008903202970307
In *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2020)*, pages 297-307
ISBN: 978-989-758-397-1; ISSN: 2184-4313

the purpose of extracting feature vectors. These conventional methods require a training data-set based on target keywords. Therefore the keyword set must be fixed. This restriction disable the methods from applying to various applications requiring customization of the keywords by users, dynamic keywords by service providers. The computational cost of using additional NNs for speaker identification (Variani et al., 2014; Chen et al., 2015b; Chen et al., 2015a; Snyder et al., 2018) is also crucial for low-resource devices, and so on.

As a different approach of speaker identification, the hierarchical multi-layer acoustic model (HiLAM) (Larcher et al., 2014) is proposed. It is based on the concept of a text-dependent hidden Markov model (HMM) with Gaussian mixture models (GMMs) for exploiting alignment information of keywords. Speaker-specific HMMs are trained using only enrollment data. Therefore, the method don't need a training data-set based on target keywords. However, in addition to calculation cost of keyword detection, the HMMs calculation cost for each speaker does not match for low-resource devices. Therefore, we focus on simple GMMs methods (Hebert and Heck, 2003; Sturim et al., 2002) in this paper.

Above described, many studies focus on speaker identification based on recognition of a specific keyword with enough data-set based on the target keyword. Further, they consider keyword detection and speaker identification separately. In this paper, we construct a consistent method of simultaneous flexible keyword detection and text-dependent speaker identification with maintaining the low computational cost. Furthermore, the method creates robust speaker models on the proposed algorithm.

The first contribution is to propose a consistent and efficient algorithm for simultaneous flexible keyword detection and text-dependent speaker identification for low-resource devices. A low-cost calculation method for flexible keyword detection is introduced using a customized Viterbi algorithm (Nasu, 2016) with NN scores. By adopting the algorithm that focuses only keyword detection, the calculation cost can be reduced compared to the conventional Viterbi algorithm. To avoid increasing computational cost for speaker identification, the proposed method employs middle-layer outputs in the flexible keyword detection NN as feature vectors. This method does not need the data-set based on the target keywords prepared in advance. GMMs are constructed using the feature vectors for registered speakers. Finally, likelihoods of keyword utterances output by the GMMs are compared.

The second contribution is to create robust speaker models on the proposed algorithm. Speaker identification using only registered utterances by users is a kind of few-shot learning, in which it is important to reduce the amount of training data without performance degradation.

During the enrollment stage for creating speaker models, a user is required to utter keywords multiple times. To reduce a user burden, the number of utterances during the enrollment stage should be minimized. Even so, the speaker models must sufficiently cover variation in the environment and user speech. To overcome the problem, the proposed method uses the DropConnect method (Wan et al., 2013) when creating speaker GMMs. Normally, DropConnect is applied to NN training for creating robust NN models. In this paper, it is used for estimating uncertainties of GMMs in speaker modeling. The proposed method augments feature vectors of middle-layer outputs using the DropConnect method to cover variation. It is based on Bayesian theory for uncertainty (Gal, 2016). Finally, speaker GMMs are created using the augmented feature vectors to estimate the uncertainty. This method does not influence the calculation cost of speaker identification process. Therefore, it can maintain low-resource-cost computation.

2 OVERVIEW OF THE PROPOSED METHOD

Figure 1 shows an overview of the enrollment process. First, each frame feature vector is input to a NN for keyword detection. This NN outputs probabilities for phoneme states and feature vectors for speaker identification on the basis of a middle layer. Recognition is based on a method introduced in Section 3 using probabilities that have been output from the NN. A feature vector from the middle layer is transformed into many feature vectors in accordance with multiple DropConnect patterns to estimate the uncertainty. Finally, speaker models are constructed using the transformed feature vectors and recognition alignment information.

In the speaker-identification step, averaged vectors for phoneme-state segments of a recognized keyword are calculated on the basis of a speaker utterance. The averaged vectors are input to each of the speaker models to estimate a likelihood score. The speaker with the highest score is selected from among the set of registered speakers as the output. A detailed diagram for speaker modeling to register speakers is shown in Fig. 2. A detailed diagram for speaker identification is shown in Fig. 3. The keyword detection

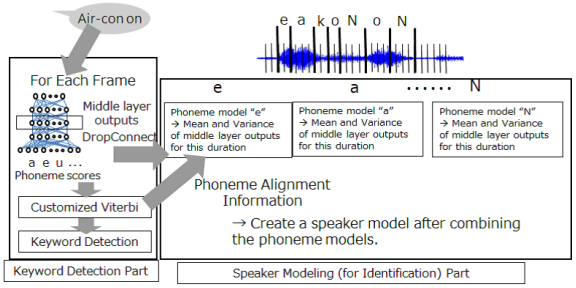

Figure 1: Overview of the proposed method.

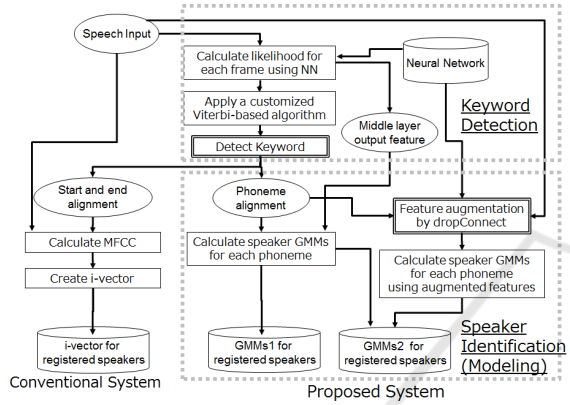

Figure 2: A detailed block diagram of the proposed method and the conventional i-vector method for speaker modeling.

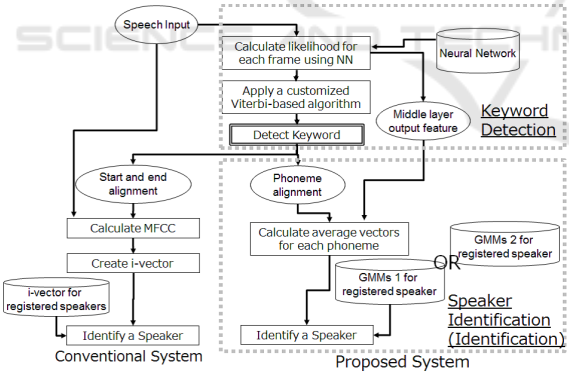

Figure 3: A detailed block diagram of the proposed method and the conventional i-vector method for speaker identification.

process is common for speaker modeling and identification. The explanation of the detection is given in Section 3, and the explanation of speaker modeling and identification is given in Section 4.

3 KEYWORD DETECTION

The process of keyword detection is shown in "Process-Detection" in Fig. 2. First, phoneme-state probability is calculated for each frame using a NN. Next, a customized Viterbi-based algorithm is applied as a flexible keyword detector.

A flexible keyword-detector with low calculation cost using phoneme-state probability was proposed in (Zhu, 2017). A score for a target keyword to be recognized is calculated by summation of phoneme-state probabilities in a sliding window. It can avoid a heavy calculation cost of a conventional Viterbi algorithm that is usually used for keyword detection (Junkawitsch et al., 1996). However, this method needs to control the size of the window, and the score is approximated.

To avoid degradation by approximation, we adopt the algorithm proposed in (Nasu, 2016) for recognition of flexible keywords. The method is based on dynamic programming using a left-to-right HMM. Keyword HMMs consist of phoneme states. T and τ denote the total number of input frames and the frame index, respectively. $\{x_\tau\}_{\tau=1}^{T}$ denotes a time sequence of speech feature vectors. N denotes the total number of keyword phoneme-states, $\{j\}_{j=1}^{N}$ denotes the state index, and $\mathrm{score}(x_\tau, q_\tau)$ denotes the local score for an assigned state q_τ in states $(1,2,..,N)$ at frame τ. Q denotes a set of $\{q_s, q_{s+1}, ..., q_e\}$. A keyword score $S(s,e)$ using HMMs for a segment from a start frame s to an end frame e is calculated using the Viterbi algorithm as follows:

$$S(s,e) = \frac{1}{e-s+1} \max_{Q} \sum_{\tau=s}^{e} \mathrm{score}(x_\tau, q_\tau). \quad (1)$$

A keyword is detected and recognized using a predefined threshold θ for a score $S(s,e)$ when a speech segment with $S(s,e) > \theta$ exists. To find the segment by the frame t ($t \in [1,T]$), the calculation is defined as follows:

$$\max_{s<t} \frac{1}{t-s+1} \max_{Q} \sum_{\tau=s}^{t} \mathrm{score}(x_\tau, q_\tau) > \theta. \quad (2)$$

The computational cost is normally $O(NT^3)$ for the calculation of many possible paths for each combination of s and e. Because the score $S(s,e)$ is not needed to know for the purpose of detection, the formula is

arranged as follows:

$$\begin{aligned} & \max_{s<t} \frac{1}{t-s+1} \max_{Q} \sum_{\tau=s}^{t} \text{score}(x_\tau, q_\tau) > \theta, \\ \Leftrightarrow \quad & \max_{s<t} \max_{Q} \sum_{\tau=s}^{t} (\text{score}(x_\tau, q_\tau) - \theta) > 0, \\ & L(s,t) > 0, \end{aligned} \tag{3}$$

where

$$L(s,t) = \max_{s<t} \max_{Q} \sum_{\tau=s}^{t} (\text{score}(x_\tau, q_\tau) - \theta).$$

This equation implies that normalization by $t-s+1$ is not needed for merely detecting a segment with $S(s,t) > \theta$. Algorithm 1 shows pseudo-code for our method. $L(s,t)$ is easily calculated by the algorithm. Each lattice point only keeps the path with the maximum summation score searched from all of the possible paths with different starting points. On the other hand, the conventional Viterbi algorithm requires control of many hypotheses having different starting points. In this method, the existence of a duration with $S(s,t) > \theta$ is guaranteed by finding a duration with $L(s,t) > 0$. Therefore, the detection performance is not degraded compared to the performance of the conventional Viterbi algorithm. However, the calculation cost is drastically reduced to $O(NT)$. In this paper, a simple and small feed forward NN is used for the calculation of a phoneme-state score.

Algorithm 1: Detector Algorithm Procedure.

The score of the max path reached for state j of frame t is $L_{t,j}$.
The max path holds alignment information.
The state score for a state j of frame t is $l_{t,j}(l_{t,j} < 0)$.
Initialize max path score $L_{0,j} = -\infty (j = 1,..,N)$ and $L_{0,0} = 0$.
$L(s,t) = L_{t,N}$. ($L(s,t)$ is the same as eq. 3.)
For frame $t (t = 1,..,T) : L_{t,0} = 0$,
Acoustic score for a state $l_{t,j}$ is calculated by a NN,
for $j = 1$ to N **do**
  if $L_{t-1,j-1} > L_{t-1,j}$ **then**
    $L_{t,j} = L_{t-1,j-1} + l_{t,j}(x_t) - \theta$
    The recorded alignment information including the starting point s of $L_{t-1,j-1}$ is propagated.
  else
    $L_{t,j} = L_{t-1,j} + l_{t,j}(x_t) - \theta$
    The recorded alignment information including the starting point s of $L_{t-1,j}$ is propagated.
  end if
end for
if $L(s,t) = L_{t,N} > 0$ **then**
  The keyword is recognized.
  Go to speaker registration or identification step.
else
  Go to $(t+1)$th iteration.
end if

4 SPEAKER IDENTIFICATION

4.1 Speaker Modeling

Three speaker modeling processes are shown in Fig. 2. One is the conventional method using i-vector, and the process is shown in "Conventional". Another is the simple GMMs method shown in "Proposal1". The other is the GMMs method using feature augmentation shown in "Proposal2". In this section, "Proposal1" is mainly introduced.

4.1.1 Feature

In the proposed method, features for speaker models are derived from a predefined middle layer in the detector NN.

One reason for using the keyword independent features is that the training set including the keywords for recognition cannot be collected prior to user-definition of keywords in our flexible keyword scenario though it is beneficial to construct robust speaker identification features such as (Variani et al., 2014; Larcher et al., 2012; Chen et al., 2015a).

Another reason is to avoid additional computation cost of other NNs for feature extraction.

However, each layer in the NN recognition step normalizes not only for channel and noise features, but also for speaker features, because the NN is trained to distinguish from among phoneme states. Channel and noise environment influences should be eliminated, but speaker identification information should be retained. Therefore, the best feature can be found in the outputs of middle layers, which are described in Section 6.

4.1.2 Modeling

A Gaussian model (GM) is constructed using features for each of the combinations of a speaker and a phoneme-state existing in a keyword. Phoneme alignment is determined as a result of keyword detection, as described in Section 3. To construct speaker GMs, mean and variance are calculated for phoneme-state segments. A lth-layer output vector is used as a feature vector. $y_{l\tau}$ denotes an output vector of the lth layer at the τth frame. W denotes the number of registered speakers. A speaker $w(0 < w \leq W)$ utters a keyword that has N HMM-states, and the recognized state sequence is Q. When the number of frames in a state $m(0 < m \leq N)$ is M_m in a state sequence Q, the

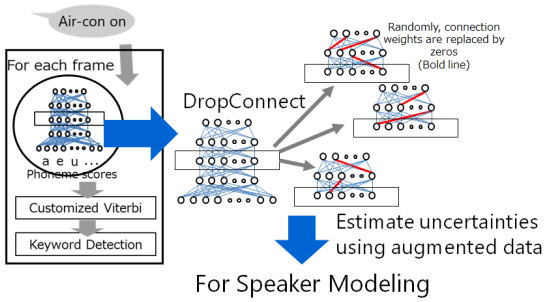

Figure 4: Overview of the DropConnect in Modeling Uncertainty of Bayesian Deep Networks method.

mean μ_{wlm} and the variance σ^2_{wlm} for the state m of a speaker w using the lth-layer output are calculated in accordance with the following equations:

$$\mu_{wlm} = \frac{1}{M_m} \sum_{\tau, q_\tau \in Q, q_\tau = m} y_{l\tau}, \quad (4)$$

$$\sigma^2_{wlm} = \frac{1}{M_m} \sum_{\tau, q_\tau \in Q, q_\tau = m} (y_{l\tau} - \mu_{wlm})^2. \quad (5)$$

For example, when the number of enrollment utterances is three, Q_1, Q_2, and Q_3 denote recognized sequences. The mean and the variance are calculated in $Q = Q_1 \cup Q_2 \cup Q_3$. To cover the small variance, "GMMs1" shown in Fig. 2 is composed of GMs created by speaker features and a global one calculated using speaker-and-keyword independent feature vectors.

4.2 Modeling Uncertainty

This section explains the GMMs method using feature augmentation shown in "Proposal2" in Fig. 2. In this process, the feature vectors of middle layer outputs from a NN for keyword detection is augmented by a DropConnect method using speech input to estimate the modeling uncertainties.

In (Gal, 2016), the relation between model uncertainty and stochastic regularization techniques such as the dropOut method, multiplicative Gaussian noise (Srivastava et al., 2014), and the DropConnect method (Wan et al., 2013) are described. In our method, this kind of technique is applied to speaker model uncertainty. The DropConnect method is chosen because it is better than the dropOut method when estimating model uncertainty in (Mobiny et al., 2019). Random layer-output is acquired through a stochastic forward pass, to which DropConnect is applied. By repeating the process R times, sampling outputs $\{\hat{y}_{l\tau 1}, ..., \hat{y}_{l\tau R}\}$ are acquired. Figure 4 shows the overview of the proposed method. In (Gal, 2016; Mobiny et al., 2019), the sampling outputs are empirical samples from an approximate predictive distribution. A predictive mean μ^*_{wlm} and variance σ^{*2}_{wlm} are approximated using Eqs. 6 and 7,

$$\mu^*_{wlm} = \frac{1}{RM_m} \sum_{\tau, q_\tau \in Q, q_\tau = m} \sum_{r=1}^{R} \hat{y}_{l\tau r}, \quad (6)$$

$$\sigma^{*2}_{wlm} = \frac{1}{RM_m} \sum_{\tau, q_\tau \in Q, q_\tau = m} \sum_{r=1}^{R} (\hat{y}_{l\tau r} - \mu^*_{wlm})^2. \quad (7)$$

The proposed method treats modeling uncertainty as generative speaker model uncertainty to support small segment data. "GMMs2" shown in Figure 2 is composed of the GMs with augmentation and the GMs in previous section.

4.3 Speaker Identification

Speaker identification process is shown in Fig. 3. The conventional i-vector process shown in "Conventional". Cosine similarity between the extracted i-vector and each registered i-vector is calculated in the process of the identification. On the other hand, the process of the proposed method is shown in "Proposal" in Fig. 3. During testing for the proposed methods, an averaged vector for each segment is calculated and input to all speaker GMMs that have been created for the keyword. Likelihoods calculated from all speaker GMMs are compared, and a speaker with the best likelihoods is selected as a result of speaker identification.

5 EXPERIMENTAL SETUP

The proposed method recognizes a keyword, and the speaker who has uttered the recognized keyword is identified from among a set of registered speakers. To this end, we have conducted two experiments on keyword detection and speaker identification.

5.1 Evaluation Database

We evaluated the proposed method using a Japanese command dataset recorded by a close-talking microphone. The sampling rate was 16000 [Hz]. For the evaluation, 39 keywords uttered 5 times by 100 speakers (female:51, male:49) were used. Ten keywords uttered 5 times by 50 speakers (female:25, male:25) were used for development. The evaluation set did not include any speakers in the development set. Table 2 and 4 show all keywords, and each averaged duration [sec] for the development and evaluation sets.

5.2 Evaluation Metrics

5.2.1 Recognition

Recognition performance was measured by the false reject (FR) rate [%] and the number of false accepts (FA). N_c and N_{all} denote the number of correctly recognized keywords and the total number of utterances, respectively. Therefore, $FR = 100 \times (N_{all} - N_c)/N_{all}[\%]$. The value of FA was calculated using noises from the third CHiME challenge (Barker et al., 2015). All noise files were concatenated together into a single file. The detector for all keywords was executed for the concatenated file, and the number of FA was calculated. Finally, the FA count was normalized by the number of hours and the number of keywords. Therefore, the metric was FA $[times/(hour \cdot \#keyword)]$.

5.2.2 Speaker Identification

For speaker identification, the evaluation metric for all experiments was identification rate (IR) value [%]. Each speaker model output a score. The scores were compared for only registered speakers. IR was calculated using the equation $\mathrm{IR} = 100 \times C/N_c$, where C was the number of utterances matched between a reference speaker and a selected speaker. There were 3 enrollment utterances for each keyword, and the remaining 2 utterances were evaluated after enrollment. The utterances were cross-validated, for a total of 5 utterances covering all possible variations. In many real use-cases of the proposed method, the number of registered speakers is less than 100. This purpose is to measure the identification performance, properly.

5.3 Model Preparation

5.3.1 Recognition

A detector was constructed using the method described in Section 3. The algorithm needed an acoustic model to calculate the acoustic score for lattice points at each frame. This system was designed to work on small devices. Therefore, the size of acoustic model was very small , and contained fewer differences in comparison with a model designed for large-vocabulary speech recognition.

The training set was the Japanese ATR database (Kurematsu et al., 1990), which included 270 hours of speech. Babble noise was added to the training data to ensure robustness. The basic characteristic was 32 Mel-filter bank features compressed to 16 dimensions using an affine transformation. Mel-filter bank features were normalized using a moving average. Input feature vectors for the DNNs were created by concatenating 10 preceding and following frames, for a total of 21 frames. Therefore, the number of dimensions was $16 \times 21 = 336$. Hidden layers were structured as $128[\mathrm{unit}] \times 4[\mathrm{layer}]$. The activation function was a sigmoid function. Final outputs comprised 4500 clustered phoneme-states.

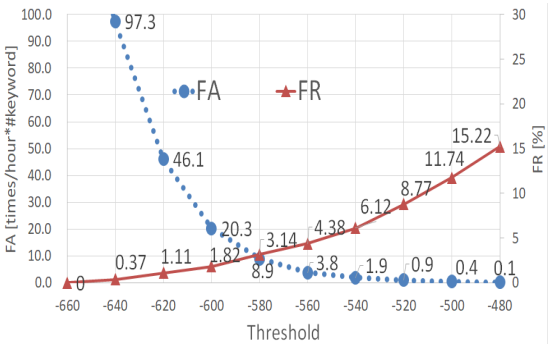

Figure 5: Detection performance FA-FR.

5.3.2 i-vector Setup for Comparison

An i-vector-based method was constructed for the comparison. The speaker-modeling process is shown in ”Conventional in Fig. 2. Features comprised 19 Mel-frequency cepstral coefficients (MFCC), in addition to energy, the Δ, and the $\Delta\Delta$. There were 60 dimensions in total. The i-vector was extracted using the ALIZE (Larcher et al., 2013) speaker identification library after normalization of features. A 2048-mixture universal background model and total variability matrix were trained using the ATR dataset, which included 3768 speakers. The number of i-vector dimensions was 200. The parameter of i-vector was similar as in (Tsujikawa et al., 2017). The speaker-identification process is shown in ”Conventional in Fig. 3. Similarity between a speaker reference i-vector and a non-reference i-vector was measured using cosine similarity. Keywords were extracted by the detector, and the extracted portions were used for enrollment and evaluation.

6 EXPERIMENT USING DEVELOPMENT SET

6.1 Recognition

The relation between FA and FR using the development set is shown in Fig. 5. The score for a threshold

in Equation 2 was calculated using accumulation of logarithm-of-output probabilities for lattice points in a detected path. The threshold in Fig. 5 was the θ in Equation 2 for all keywords. In all experiments, a threshold of -540 was selected to avoid FA in a noisy environment. Therefore, the total value of the speaker-identification target for the development set was $10(\text{keywords}) \times 50(\text{speakers}) \times {}_5C_2 \times 2 \times (1 - 0.0612) = 9388$. The threshold was decreased by 10 until the keyword was recognized during enrollment.

6.2 Speaker Identification

6.2.1 Euclidean Distance and Simple GMMs

We first conducted experiments in which we measured the Euclidean distances between vectors of phoneme-state features, as in (Dutta, 2007). The detector output phoneme-state alignment, and averaged vectors were calculated for each phoneme-state using Eq. 4. For enrollment, vectors for each phoneme-state segment of three utterances were averaged to create a speaker vector. The speaker vectors were compared to evaluation vectors from test utterances in accordance with their Euclidean distances. The speaker with the speaker vector closest to an evaluation vector was selected as an output speaker. A feature vector was extracted from each middle layer of the detector NN before application of the sigmoid activation function. The structure of the hidden layers was $128[\text{unit}] \times 4[\text{layer}]$ for the detector. Therefore, the number of dimensions was 128 for a feature vector, and 4 patterns from 4 layer-outputs were obtained.

"Euclid" in Tab. 1 represents the performance. "Layer" means the use of the (1st, 2nd, 3rd, 4th) layer output vector. Performance of the i-vector method was 76.10%. Performance of the simple Euclidean distance method was better than that of the i-vector method. Therefore, alignment information has been demonstrated to be important for speaker identification using short keywords.

Next, the averaged vectors were replaced by GMMs. The process is shown in "Proposal1" in Fig. 2. An averaged vector for each phoneme-state was input to the speaker-GMMs that had been constructed from phoneme-state GMMs, and each of the speaker-GMMs output a likelihood value for an utterance to be recognized. To prevent an extremely low score, GMMs were composed of a GM created by speaker features and a global one. A global GM was created in advance using training data. Weights were equal for the two GMs. The output likelihoods were compared to identify a speaker. For compensation, a global variance σ_{lg}^2 and σ_{wlm}^{*2} were summed after weighting. A global variance σ_{lg}^2 was calculated using the training dataset.

Table 1 shows the simple GMM-based method (GMM(0)). Clearly, the performance of the GMMs was better than that of Euclid. Therefore, variance was effective for the identification, even though there were few enrollment data values.

6.2.2 GMMs using Augmented Feature Vectors

In this work, GMs created by augmented vectors were added to the original GMMs described in Section 6.2.1. The process is shown in "Proposal2" in Fig. 2. In this manner, three GMMs having equal weights were constructed when augmented vectors were applied. Feature vectors for speaker identification were augmented using DropConnect. DropConnect has a parameter "drop rate" that can control the rate of dropping connections of a NN. As a controlled parameter, the number of augmentation iterations R was also considered. We controlled the values of the drop rates for all iterations simultaneously. For each iteration, the best drop rate was selected from among $(10, 20, 30, 40, 50, 60)$%. The parameter set was one for all keywords, and was not tuned for individual keywords.

Table 1 shows the results when the parameters were controlled. ( ) shows the number of iteration R. It shows that augmentation was effective for all layers. It shows the best performance for each layer after controlling the drop rates. The best performance was 91.65%, using the 4th-layer output in one-layer output results. Without augmentation, the performance of the first-layer output was the best. The first-layer output included much speaker information, but also other noise information. On the other hand, deeper layers tended to eliminate other information that had been excluded for phoneme-state discrimination. Therefore, augmentation was capable of compensating for inner-speaker variation to maintain the robustness even in the presence of other factors. Next, outputs of both the first and the fourth layer were used to exploit both features. Performance was represented in Tab. 1 "1st+4th". The performance was 92.66% when the number of iterations was 10. Table 2 shows speaker identification performance for each keyword using the development set.

7 EXPERIMENT USING EVALUATION SET

We conducted the experiment using the evaluation set. The parameters controlled using the development set

Table 1: *Performance using the development set by IR [%].*

i-vector	76.10					
		GMMs (#Iteration)				
Layer	Euclid	(0)	(3)	(5)	(10)	(12)
1st	80.39	88.45	89.49	90.53	90.51	90.48
2nd	80.13	87.44	88.76	89.29	90.03	90.18
3rd	81.76	86.56	88.82	89.74	90.32	90.41
4th	84.42	87.87	90.29	91.07	91.65	91.63
1st+4th	*	88.70	91.33	92.22	92.66	92.72

Table 2: *Performance using the development set by IR [%].*

Keyword	Duration [sec]	i-vector	GMM(0) 1st+4th	GMM(10) 1st+4th
u shi ro	0.29	39.45	80.22	88.40
ja N pu	0.32	72.55	78.18	85.07
mo do ru	0.34	68.24	79.68	86.63
i chi ba N	0.40	87.84	85.88	90.1
kya N se ru	0.46	70.77	95.50	96.79
to ri ke shi	0.47	91.42	95.29	96.13
ri se Q to	0.48	82.57	81.29	88.65
ko ma N do	0.54	74.95	94.81	97.96
tu gi no pe e ji	0.74	86.61	97.98	97.98
ma e no pe e ji	0.75	85.08	96.95	98.21

Table 3: *Performance using the evaluation set by IR [%].*

i-vector	71.22				
Layer	1st	2nd	3rd	4th	1st+4th
GMM(0)	86.39	85.21	83.88	84.51	85.87
GMM(10)	88.03	87.10	86.75	88.18	89.29

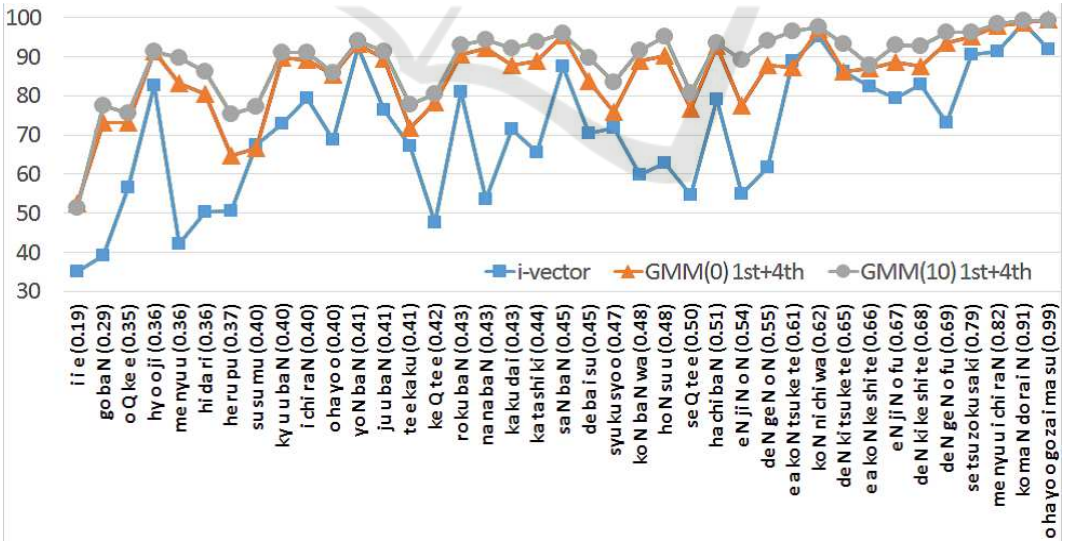

Figure 6: Speaker identification performance IR [%] using the evaluation set, and ( ) shows the averaged duration for each keyword.

were used for the evaluation. FR was 8.21%, and FA was 0.6 [times/hour · #keyword]. Speaker identification was applied to the recognized keywords. Table 3 shows the speaker identification performance. The i-vector performance for speaker identification was 71.22%.In contrast, the performance of the pro-

Table 4: *Speaker identification performance using the evaluation set by Identification Rate [%].*

Modeling		i-vector	GMM(0)	GMM(10)	GMM(0)	GMM(10)
Feature		MFCC	1st+4th layer		1st+4th layer	
#enrollment utterances		3	3		2	
Keyword	Dur. [sec]					
i i e	0.19	35.10	52.40	51.29	44.46	44.26
go ba N	0.29	39.11	73.10	77.44	67.30	72.58
o Q ke e	0.35	56.60	73.27	75.54	66.05	68.89
hy o o ji	0.36	82.75	91.40	91.40	86.40	87.10
me nyu u	0.36	42.06	83.24	89.65	76.59	85.24
hi da ri	0.36	50.35	80.58	86.05	72.52	80.07
he ru pu	0.37	50.62	64.77	75.43	56.92	67.57
su su mu	0.40	67.56	66.70	77.34	58.54	69.71
kyu u ba N	0.40	72.75	89.63	91.05	84.04	85.51
i chi ra N	0.40	79.31	89.14	90.98	83.03	85.37
o ha yo o	0.40	68.66	85.48	86.04	78.78	80.88
yo N ba N	0.41	92.30	93.31	94.01	88.33	89.24
ju u ba N	0.41	76.48	89.38	91.30	84.51	87.41
te e ka ku	0.41	76.48	89.38	91.30	64.68	72.17
ke Q te e	0.42	47.51	78.22	80.59	69.54	73.24
ro ku ba N	0.43	80.92	90.54	92.92	84.54	88.68
na na ba N	0.43	53.66	92.17	94.46	86.01	90.03
ka ku da i	0.43	71.57	87.82	92.26	83.24	88.74
ka ta shi ki	0.44	65.40	89.02	93.78	81.23	89.23
sa N ba N	0.45	87.57	95.32	96.07	90.84	92.05
de ba i su	0.45	70.35	83.86	89.64	75.18	82.33
syu ku syo o	0.47	71.74	75.77	83.34	70.17	78.70
ko N ba N wa	0.48	59.82	88.77	91.69	84.74	86.76
ho N su u	0.48	62.79	90.38	95.16	84.50	92.57
se Q te e	0.50	54.69	76.75	80.71	70.70	74.92
ha chi ba N	0.51	79.04	92.73	93.54	88.23	89.65
e N ji N o N	0.54	55.00	77.56	89.04	72.44	84.94
de N ge N o N	0.55	61.71	87.88	94.06	84.85	91.38
e a ko N tsu ke te	0.61	89.01	87.29	96.56	81.88	93.39
ko N ni chi wa	0.62	95.27	96.93	97.48	95.07	95.27
de N ki tsu ke te	0.65	86.23	86.23	93.35	83.02	88.19
e a ko N ke shi te	0.66	82.26	87.07	87.74	95.19	96.48
e N ji N o fu	0.67	79.32	88.58	93.03	86.43	90.44
de N ki ke shi te	0.68	83.02	87.46	92.63	84.33	88.14
de N ge N o fu	0.69	73.12	93.53	96.14	90.12	91.76
se tsu zo ku sa ki	0.79	90.51	95.18	96.30	92.12	93.57
me nyu u i chi ra N	0.82	91.26	97.84	98.53	93.95	95.74
ko ma N do ra i N	0.91	98.78	98.58	99.14	97.16	97.67
o ha yo go za i ma su	0.99	91.76	99.39	99.34	97.78	97.37
Average	0.51	71.22	85.87	89.29	80.96	85.11

posed method was 89.29% when the feature combination of ”1st” and ”4th” layer output (1th+4th) and 10 augment iterations (GMM(10)) were applied.

Figure 6 and Tab. 4 show speaker identification performance for each keyword using the evaluation set.

In Fig. 6, value in parentheses after each keyword shows the averaged duration of the keyword. The

order was sorted by the duration. The average duration of all keywords was 0.51. "i-vector (3utterances)" shows i-vector results using three enrollment utterances. "GMM(0)1st+4th (3utterances)" and "GMM(10)1st+4th (3utterances)" show the result of GMM(0) and the result of GMM(10) using three enrollment utterances, respectively. "GMM(0)1st+4th (2utterances)" and "GMM(10)1st+4th (2utterances)" show the result of GMM(0) and the result of GMM(10) using two enrollment utterances, respectively.

Table 4 shows the values of the results. Basically, the performance increased as the duration was longer. The performances of GMM(10) using two and three enrollment utterances were almost over 90% for more than 0.5 [sec] duration, and they were higher than that of i-vector using three enrollment utterances even if GMM(10) used two enrollment utterances.

Figure 6 and Tab. 4 also show that the augmentation was effective for most of evaluation keywords. The performance of the proposed method improved in comparison with the i-vector method. Furthermore, the augmentation method increased performance. The results show that our proposed method was effective for speaker identification using short keywords.

8 CONCLUSION

This paper proposed a speaker identification method for even a very short duration of keywords recognized by a NN-based detector. Because the feature of speaker identification is keyword independent, the proposed method can be used for various applications using flexible keywords. Moreover, computation cost for speaker identification is very small because the feature is derived from the NN of the detector without any additional NNs. The identification rate when using a conventional i-vector method was 71.22%. In contrast, the performance of the proposed method was 89.29% while maintaining low-resource-cost computation. Performance of the proposed method was clearly better than that of the conventional i-vector method for speaker identification using short keywords and few enrollment data values with a low-resource computation cost.

REFERENCES

Barker, J., Marxer, R., Vincent, E., and Watanabe, S. (2015). The third 'chime'speech separation and recognition challenge: Dataset, task and baselines. In *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 504–511. IEEE.

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

Chen, N., Qian, Y., and Yu, K. (2015a). Multi-task learning for text-dependent speaker verification. In *Proceedings of the 16th Annual Conference of the International Speech Communication Association*, pages 185–189. International Speech Communication Association (ISCA).

Chen, Y.-h., Lopez-Moreno, I., Sainath, T. N., Visontai, M., Alvarez, R., and Parada, C. (2015b). Locally-connected and convolutional neural networks for small footprint speaker recognition. In *Proceedings of the 16th Annual Conference of the International Speech Communication Association*, pages 1136–1140. International Speech Communication Association (ISCA).

Dehak, N., Kenny, P. J., Dehak, R., Dumouchel, P., and Ouellet, P. (2011). Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798.

Dutta, T. (2007). Text dependent speaker identification based on spectrograms. *Proceedings of Image and vision computing*, pages 238–243.

Gal, Y. (2016). *Uncertainty in deep learning*. PhD thesis, University of Cambridge.

Hebert, M. and Heck, L. (2003). Phonetic class-based speaker verification. In *Eurospeech 2003 - Interspeech 2003*. ISCA.

Junkawitsch, J., Neubauer, L., Hoge, H., and Ruske, G. (1996). A new keyword spotting algorithm with pre-calculated optimal thresholds. In *Proceeding of Fourth International Conference on Spoken Language Processing. ICSLP'96*, volume 4, pages 2067–2070. IEEE.

Kanagasundaram, A., Vogt, R., Dean, D. B., Sridharan, S., and Mason, M. W. (2011). I-vector based speaker recognition on short utterances. In *Proceedings of the 12th Annual Conference of the International Speech Communication Association*, pages 2341–2344. International Speech Communication Association (ISCA).

Kurematsu, A., Takeda, K., Sagisaka, Y., Katagiri, S., Kuwabara, H., and Shikano, K. (1990). Atr japanese speech database as a tool of speech recognition and synthesis. *Speech communication*, 9(4):357–363.

Larcher, A., Bonastre, J.-F., Fauve, B. G., Lee, K.-A., Lévy, C., Li, H., Mason, J. S., and Parfait, J.-Y. (2013). ALIZE 3.0 - open source toolkit for state-of-the-art speaker recognition. In *Proceedings of the 14th Annual Conference of the International Speech Communication Association*, pages 2768–2772.

Larcher, A., Bousquet, P.-M., Lee, K. A., Matrouf, D., Li, H., and Bonastre, J.-F. (2012). I-vectors in the context of phonetically-constrained short utterances for speaker verification. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2012*, pages 4773–4776. IEEE.

Larcher, A., Lee, K. A., Ma, B., and Li, H. (2014). Text-dependent speaker verification: Classifiers, databases and rsr2015. *Speech Communication*, 60:56–77.

Mobiny, A., Nguyen, H. V., Moulik, S., Garg, N., and Wu, C. C. (2019). Dropconnect is effective in modeling uncertainty of bayesian deep networks. *arXiv preprint arXiv:1906.04569*.

Nasu, Y. (2016). Detection apparatus, detection method, and computer program product. US Patent App. 15/071,669.

Poddar, A., Sahidullah, M., and Saha, G. (2015). Performance comparison of speaker recognition systems in presence of duration variability. In *India Conference (INDICON), 2015 Annual IEEE*, pages 1–6. IEEE.

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., and Khudanpur, S. (2018). X-vectors: Robust DNN embeddings for speaker recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, pages 5329–5333.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Sturim, D. E., Reynolds, D. A., Dunn, R. B., and Quatieri, T. F. (2002). Speaker verification using text-constrained gaussian mixture models. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pages I–677. IEEE.

Tsujikawa, M., Nishikawa, T., and Matsui, T. (2017). I-vector-based speaker identification with extremely short utterances for both training and testing. In *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, pages 1–4. IEEE.

Variani, E., Lei, X., McDermott, E., Moreno, I. L., and Gonzalez-Dominguez, J. (2014). Deep neural networks for small footprint text-dependent speaker verification. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014*, pages 4052–4056. IEEE.

Wan, L., Zeiler, M., Zhang, S., Le Cun, Y., and Fergus, R. (2013). Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1058–1066.

Wu, M., Panchapagesan, S., Sun, M., Gu, J., Thomas, R., Vitaladevuni, S. N. P., Hoffmeister, B., and Mandal, A. (2018). Monophone-based background modeling for two-stage on-device wake word detection. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2018*, pages 5494–5498. IEEE.

Zhu, L. (2017). Neural network-based small-footprint flexible keyword spotting. Master's thesis, University of Karlsruhe Institute of Technology.