A low latency technique for speaker detection from a large negative list

Yu Zhou
Vail Systems, Inc.
yzhou@vailsys.com

Vijay K. Gurbani
Vail Systems, Inc.
vgurbani@vailsys.com

B. Chandra Mouli
Vail Systems, Inc.
chandra@vailsys.com

Abstract

Negative list (NL) detection, commonly referred as open-set multi-target speaker detection, attempts to match a test utterance with any one of a set of known utterances enrolled in the negative list. A number of normalization techniques have been developed for similarity score calibration in order to increase the detection accuracy. While these normalization methods apply to both single-target verification and multi-target detection, in this work we propose NL-Norm, a novel normalization technique that is designed specifically for multi-target detection by considering scores between all enrolled NL utterances and the normalization cohort as a single distribution. Furthermore, we propose using Locality Sensitive Hashing (LSH) to efficiently find a small subset of utterances from enrolled NL utterances and the normalization cohort that are most similar to the test utterance, so that the number of similarity score computations can be significantly reduced. The combination of these novel techniques is evaluated on the MCE 2018 datasets. Applying LSH and NL-Norm, our approach demonstrated significant improvements in both speed and accuracy over using PLDA only in the backend, resulting in 88% reduction in detection time while decreasing the equal-error rate (EER) from 6.49% to 5.48% for MCE2018 dataset.

1 Introduction

Much progress has been made in speaker recognition, as demonstrated by continuously improving results in US National Institute of Standards and Technology (NIST) Speaker Recognition Evaluation series (Greenberg et al., 2020; Sadjadi et al., 2020). Meanwhile, in real world applications such as call center services, Negative List (NL) detection is often a crucial component of fraud detection based on voice biometrics. In NL detection, a call is flagged for investigation if an utterance is determined to be spoken by one of the known fraudulent speakers that are enrolled in the NL, without needing to identify which specific NL speaker the caller is matched with.

NL detection is often referred as open-set multi-target speaker detection. While most speaker recognition studies focus on the single-target problem, in recent years there have been efforts to develop methods used for multi-target speaker recognition (Singer and Reynolds, 2004; Zigel and Wasserblat, 2006; Malegaonkar and Ariyaeeinia, 2011; Gunson et al., 2015) For example, MCE 2018 (Shon et al., 2019) is a challenge designed specifically to promote methods for multi-target cohort detection and multi-target identification: the former, also known as Top-S detction, is aimed to only detect whether the input speech is spoken by a member of the NL cohort; the latter, referred as Top-1 detection, not only detects membership in the NL cohort but further identifies the specific speaker within the NL. In this paper we focus on the Top-S detection only, and use the call center industry term Negative List detection to distinguish it from Top-1 detection or open-set speaker identification.

The techniques used by NL detection are mostly common with that of single-target speaker recognition and are depicted in Figure 1. During the training phase (Figure 1(a)), a front-end Gaussian Mixture Model with Universal Background Model (GMM/UBM) (Reynolds et al., 2000; Dehak et al., 2011)) or a Deep Neural Network (DNN) model

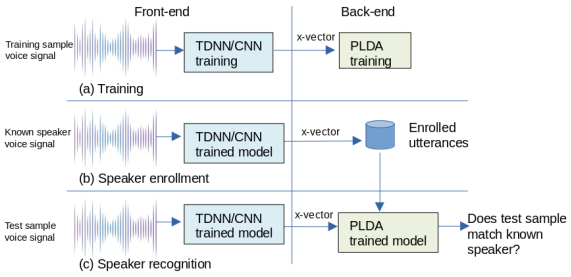

Figure 1: Steps in speaker identification

(Snyder et al., 2018) is trained to extract the speaker embedding i-vector or x-vector, respectively. (For brevity, Figure 1 only shows x-vector.) Subsequently, a back-end model such as Probabilistic Linear Discriminant Analysis (PLDA) is trained to produce a matching score between a pair of input i-vectors or x-vectors representing the log-likelihood ratio of the two vectors belonging to the same speaker.

Next, during the speaker enrollment phase (Figure 1(b)), i-vectors or x-vectors are extracted from a known speaker's utterances and stored in the database as voice biometric signature of that speaker. Finally, for speaker recognition (Figure 1(c)), the i-vector or x-vector extracted from an out-of-sample test utterance is paired with the enrolled i-vector or x-vector of a known speaker for the PLDA model to compute the matching score in order to determine whether the test utterance matches the enrolled speaker (Prince and Elder, 2007). Score normalization (Fortuna et al., 2004, 2005; Zigel and Wasserblat, 2006; Matějka et al., 2017) is typically performed on the PLDA output before it is compared with a pre-determined threshold for the match/no-match decision.

Since single-target and multi-target speaker recognitions share the same techniques described above, a multi-target speaker recognition problem can be effectively treated as multiple single-target recognitions. However, for call center services, there are often thousands of fraudulent speakers in the NL, which poses some unique challenges: with the increase in the NL size, the detection error as measured by equal error rate (EER) becomes higher (Shon et al., 2019), and the computing cost in the form of detection latency for each test grows as well. These challenges impede an effective real-time NL detection implementation. In this work, we propose enhancements to back-end processing for multi-target applications with the twin objectives of lowering EER and achieving a faster time for NL detection in the face of an increasing number of speakers in the NL list.

Our contributions are:

- **A normalization technique (NL-Norm) to calibrate consistent scores for NL detection.** We describe a normalization technique that considers similarity scores between the normalization cohort and all enrolled NL speakers as a single distribution; this helps to better calibrate test scores and select a consistent threshold for NL detection;
- **Reduction of the computation cost at inference time.** Locality Senstive Hashing (LSH (Indyk and Motwani, 1998)) is used to find a subset of NL speakers and a subset of the normalization cohort that are most similar to the test utterance, so that the similarity scores are evaluated between the test utterance and these two subsets only, which reduces the computation cost at inference time. The advantage of LSH is further amplified when different adaptive lengths for Z-Norm and T-Norm are allowed within AS-Norm.

2 Preliminaries

In existing literature, the PLDA model is often combined with score normalization to create an effective backend processing of a speaker recognition system (Fortuna et al., 2004, 2005; Zigel and Wasserblat, 2006; Matějka et al., 2017). Once the PLDA score, $s(e,t)$, between a test utterance t and an NL-enrolled utterance e is generated, various normalization techniques can be applied for score calibration in order to derive a consistent matching threshold. For NL detection, t is determined as an NL match if the highest normalized score between t and all NL utterances is above the threshold; this results in a Top-S set from which the EER metric is drawn to measure the accuracy of the model (Shon et al., 2019; Singer and Reynolds, 2004).

2.1 Score Normalization

All score normalizations require a normalization cohort consisting utterances from speakers that are neither in NL nor part of the test cohorts. We denote NL-enrolled utterance set and the normalization cohort as $\mathcal{E}$ and $\mathcal{C}$, respectively:

$$\begin{aligned} \mathcal{E} &= \{e_i \mid 1 \le i \le E\} \\ \mathcal{C} &= \{c_i \mid 1 \le i \le N\} \end{aligned} \tag{1}$$

where E and N are the size of NL and the normalization cohort, respectively.

Z-Norm (Li and Porter, 1988) utilizes the scores between an NL speaker utterance e and every utterance c_i in the normalization cohort:

$$S_e = \{s(e,c_i) \mid 1 \le i \le N\} \tag{2}$$

resulting in the normalized score:

$$s(e,t)_{z-norm} = \frac{s(e,t) - \mu(S_e)}{\sigma(S_e)} \tag{3}$$

where $\mu(S_e)$ and $\sigma(S_e)$ are the mean and standard deviation of S_e, respectively. For NL detection, Eq.(2) needs to be evaluated for every enrolled speaker $e \in \mathcal{E}$. However, these evaluations can be carried out during NL enrollment instead of at inference time. At each NL detection, the most computationally expensive task is to calculate E similarity scores $\{s(e_i, t) \mid 1 \leq i \leq E\}$, which is required regardless of score normalization.

T-Norm (Auckenthaler et al., 2000) uses the scores between t and every utterance c_i in the normalization cohort:

$$S_t = \{s(t, c_i) \mid 1 \leq i \leq N\}$$
$$s(e,t)_{t-norm} = \frac{s(e,t) - \mu(S_t)}{\sigma(S_t)} \quad (4)$$

In contrast to Z-Norm, Eq.(4) is evaluated at inference time for a given t, therefor there are $(N + E)$ similarity scores to be computed at each NL detection using T-Norm.

AS-Norm (Karam et al., 2011; Cumani et al., 2011) is often found to have the best performance, especially for multi-target recognitions. It is defined as the average of adaptive Z-Norm and T-Norm, namely, for an adaptive length K:

$$s(e,t)_{as-norm} = \frac{1}{2}\left(\frac{s(e,t) - \mu(S_e^{(K)})}{\sigma(S_e^{(K)})} + \frac{s(e,t) - \mu(S_t^{(K)})}{\sigma(S_t^{(K)})}\right) \quad (5)$$

where $S_e^{(K)}$ and $S_t^{(K)}$ denote the subsets consisting of the highest K scores in S_e and S_t, respectively. For NL detection, S_e (thus $S_e^{(K)}$) can be evaluated as soon as the NL is constructed, however S_t (thus $S_t^{(K)}$) can only be calculated when the test utterance t is present. Therefore at inference time for each t the number of similarity scores to be generated is $(N + E)$.

2.2 MCE 2018 dataset description

The work reported in this paper is conducted on the MCE 2018 dataset, a public dataset curated from recordings of customer-agent conversations to an operational call center (Shon et al., 2019). The dataset, consisting of negative list speakers and background speakers (i.e. not on the negative list), is summarized in Table 1.

The MCE 2018 dataset is provided in the form of i-vectors corresponding to each of the negative list and background speaker utterances. Using this dataset has several advantages: NL detection (Top-S detection) is one of the tasks in MCE challenge;

Table 1: MCE 2018 dataset description

Set	Subset	No. of speakers	Total utterances
Train	Negative list	3,631	10,893
	Background	500	30,952
Dev.	Negative list	3,631	3,631
	Background	5,000	5,000
Test	Negative list	3,631	3,631
	Background	12,386	12,386

the dataset consists of 600-dimension i-vectors extracted from call center conversations, the domain of interest in our study; and the large number of enrolled NL speakers E = 3631 is within range of real-world NL sizes. In addition, the techniques investigated in this work are part of the speaker recognition backend process, therefore using a set of i-vectors that has been validated by previous studies (Khoury et al., 2019; Font, 2019; Wilkinghoff, 2020) eliminates the need for vector extraction, and prevents introducing unnecessary variabilities for the purpose of this work.

2.3 Using LSH for low-latency search at inference time

PLDA model is the preferred method for similarity score generation due to its accuracy as measured in EER (Prince and Elder, 2007; Matejka et al., 2011). However, PLDA is computationally expensive for NL detection as it requires computing a large number of pairwise scores to determine the membership of a test utterance in the NL set. While Linear Discriminant Analysis (LDA) (Matejka et al., 2011) can be applied prior to PLDA to reduce dimensions and speed up computation of the score, the limiting factor will be the size of the NL. In commercial call-center applications it is not uncommon for the NL to contain thousands of entries, making the PLDA approach unfeasible for real-time detection.

To speed up the search, we propose using LSH, a family of functions used to solve the nearest neighbor problem by finding approximate — instead of exact — matches (Indyk and Motwani, 1998). LSH hashes the data and a query point in a way that maximizes the probability of a collision for points that are close to each other than for those which are farther apart. This approximation allows efficient solutions to exist when the dimensionality, m, is large. Further, Indyk and Motwani (Indyk and Motwani, 1998) show that LSH requires $O(mn^{1+1/c})$ processing time and $O(mn^{1/c})$ query time, where c is a constant that constrains the radius around which points should match. A crucial parameter in LSH is the choice of a distance function; exist-

ing literature (Charikar, 2002; Schmidt et al., 2014) demonstrates that the cosine similarity measure can be approximated well with locality sensitive hash functions.

LSH minimizes run time at the expense of accuracy, however, as our results in Section 6 demonstrate, the approximate matches retrieved by LSH have a high probability of being correct as reflected in the lowered EER.

3 Related Work

Open-set speaker recognition techniques using PLDA are often enhanced with score normalization. Li and Porter (Li and Porter, 1988) propose Z-Norm, which normalizes the PLDA score between the test and target utterances to the distribution of scores between the target and a normalization cohort. Auckenthaler et al. (Auckenthaler et al., 2000) present T-Norm, which applies the score distribution between the test utterance and the normalization cohort. S-Norm (Kenny, 2010) is defined as the average of Z-Norm and T-Norm, while Karam et al. (Karam et al., 2011) and Cumani et al. (Cumani et al., 2011) suggest AS-Norm, which constrains the S-Norm computations to subsets that contain the highest normalization scores from the T-Norm and the Z-Norm in order to produce the most relevant PLDA score distributions (Eq.(5)).

Our contribution, NL-Norm, is fashioned after AS-Norm with one distinction unique to the Negative List detection. Instead of normalizing to the score distribution of a single target utterance as Z-Norm, NL-Norm constructs the normalization distribution using the collection of PLDA scores between the normalization cohort and all enrolled utterances in the NL. In addition, by allowing different adaptive lengths for T-Norm and (modified) Z-Norm terms in AS-Norm and NL-Norm, we can take full advantage of LSH for optimal speed and accuracy in NL detection. The detailed description is provided in Section 4.

There exists a large body of literature on the use of LSH for audio data, especially in discovering similar songs and de-duplicating remixes. However, our literature review restricts itself to those works that use LSH in the context of NL detection. The most closely related work on using LSH for NL detection is Schmidt et al. (Schmidt et al., 2014), which uses LSH to quickly approximate the cosine distance in their retrieval process. They eschew the use of PLDA because "this method performs a more complicated hypothesis for i-vector matching, which impedes its use with LSH." (Schmidt et al., 2014). While the computationally expensive nature of PLDA precludes its use as a distance function in LSH, our work demonstrates that PLDA can nonetheless be used effectively by allowing LSH to constrain the number of PLDA operations required to determine a match from the NL. Ma et al. (Ma et al., 2015) create clusters of low dimensional i-vectors and restrict PLDA score computation to the top-n closest clusters. They compare their approach to LSH, but unlike our work, they do not use LSH to restrict the PLDA score computations to a small set. Other surveyed works use LSH, however, unlike our work, they do not consider in in the context of PLDA computations: Jeon et al. (Jeon and Cheng, 2012) use kernelized LSH for speaker identification, while Leary et al. (Leary and Andrews, 2014) substitute the Hamming distance for cosine distance in LSH; and finally, Shon et al. (Shon et al., 2018) use random projects generated from a speaker variability space to derive the characteristic matrix in LSH.

4 Negative List Specific Techniques

In this section we propose novel techniques devised for NL detection, including: using NL-Norm, a NL-specific score normalization, to improve detection accuracy; taking advantage of different adaptive lengths for S_e and S_t in AS-Norm (Eq.(5)) and NL-Norm; and applying LSH in conjunction with NL-Norm to reduce detection latency at inference time.

4.1 NL-Norm

Z-Norm in Eq.(3) is designed to compensate similarity score variations against a single target speaker e, so that the scores between e and all test utterances can be normalized to the same distribution in order to apply a consistent threshold for speaker verification. For NL detection, since all enrolled speaker utterances are present at testing time, and a single threshold for normalized scores is needed for all speakers in NL, it is reasonable to introduce a normalization over the entire NL cohort which we refer as NL-Norm. Formally, the scores used for NL-Norm consists of pairwise scores between every normalization cohort member c_i and every NL member e_j:

$$S_{NL} = \{S_{e_j} \mid 1 \leq j \leq E\} \tag{6}$$

where S_{e_j} is a rewrite of Eq.(2):

$$S_{e_j} = \{s(e_j, c_i) \mid 1 \leq i \leq N\} \tag{7}$$

Practically, we define NL-Norm with an adaptive length K that is analogous to AS-Norm in Eq.(5):

$$s(e,t)_{nl-norm} = \frac{1}{2}\left(\frac{s(e,t)-\mu(S_{NL}^{(K)})}{\sigma(S_{NL}^{(K)})} + \frac{s(e,t)-\mu(S_t^{(K)})}{\sigma(S_t^{(K)})}\right) \quad (8)$$

where

$$S_{NL}^{(K)} = \{S_{e_j}^{(K)} \mid 1 \le j \le E\} \quad (9)$$

with $S_{e_j}^{(K)}$ denoting the subset consisting of the highest K socres in S_{e_j}.

4.2 Different Adaptive Lengths

Even though Eq.(5) is commonly used for AS-Norm, there is no theoretical constraint that the same adaptive length K must be used for Z-Norm and T-Norm terms. By allowing different adaptive lengths K_e and K_t for Z-Norm and T-Norm, respectively, Eq.(5) can be modified as:

$$s'(e,t)_{as-norm} = \frac{1}{2}\left(\frac{s(e,t)-\mu(S_e^{(K_e)})}{\sigma(S_e^{(K_e)})} + \frac{s(e,t)-\mu(S_t^{(K_t)})}{\sigma(S_t^{(K_t)})}\right) \quad (10)$$

Similarly, Eq.(8) for NL-Norm can be modified with different adaptive lengths K_t and K_e:

$$s'(e,t)_{nl-norm} = \frac{1}{2}\left(\frac{s(e,t)-\mu(S_{NL}^{(K_e)})}{\sigma(S_{NL}^{(K_e)})} + \frac{s(e,t)-\mu(S_t^{(K_t)})}{\sigma(S_t^{(K_t)})}\right) \quad (11)$$

While employing different K_e and K_t in AS-Norm is not a novel approach (it was previously used in (Wilkinghoff, 2020)), here we explicitly explore its advantages when applied to NL detection: decoupling K_e and K_t not only enables further optimization of detection accuracy, it also allows the selection of a small K_t value without sacrificing the benefit of Z-Norm that may require a larger K_e. The combination of LSH with a small K_t can significantly improve the speed of NL detection by reducing the number score computations between the test utterance t and the normalization cohort, which is described further in the next section.

4.3 Reducing inference latency through LSH

In NL detection, for a given test utterance t, its similarity scores with all members of the NL are computed and ranked, with or without score normalization. The top ranked score is then compared with the pre-determined threshold to reach a decision. The search time here will be dominated by $O(E)$, where $E = |\mathcal{E}|$.

As discussed in Section 2.3, LSH is used to speed up the search process. For the work described here, this means the following: First, the LSH pipeline is "trained" on the i-vectors or x-vectors associated with the $\mathcal{E}$ and $\mathcal{C}$. (Recall from Section 2 that $\mathcal{C}$ is the normalization cohort.) Here, "training" implies deriving shorter characteristic vector for each of the 600-dimension vectors in $\mathcal{E}$ and $\mathcal{C}$ using a set of random hyperplane-based hash functions (Charikar, 2002). Given a collection of vectors in $\mathbb{R}^m$, we choose a random vector $\vec{r}$ from the $m-$dimension Gaussian distribution and define a hash function $h_{\vec{r}}$ as follows:

$$h_{\vec{r}}(\vec{u}) = \begin{cases} 1 & \vec{r}\cdot\vec{u} \ge 0 \\ 0 & \vec{r}\cdot\vec{u} < 0 \end{cases} \quad (12)$$

Then, for any vectors $\vec{u}$ and $\vec{v}$,

$$\mathbf{Pr}[h_{\vec{r}}(\vec{u}) = h_{\vec{r}}(\vec{v})] = 1 - \frac{\theta(\vec{u},\vec{v})}{\pi} \quad (13)$$

where $\theta(\vec{u},\vec{v})$ is the angle between vectors u and v. Further, for n vectors, the hash functions can be chosen by picking $O(log^2 n)$ random bits, thereby restricting the random hyperplanes to be in a family of size $2^{O(log^2 n)}$ (Charikar, 2002). For a given test i-vector t, we use LSH twice: once to discover K_e nearest neighbors to t from $\mathcal{E}$, and the second time to discover K_t nearest neighbors to t from $\mathcal{C}$ (the K_t mentioned in Section 4.2). Therefore, in Eq.(10) and Eq.(11), $S_t^{(K_t)}$ is constructed by identifying K_t members in $\mathcal{C}$ using LSH followed by the generation of K_t scores, instead of generating all N scores followed by the identification of top K_t scores. (Recall that $N = |\mathcal{C}|$.) Using such an approach, for each t, the number of PLDA score evaluations is reduced from $O(E+N)$ to $O(K_e + K_t)$. Because LSH search time is negligible comparing with PLDA score evaluation, this approach can significantly reduce the computational cost and latency of NL detection.

5 Experimental Setup

The training set of MCE dataset, consisting of 3,631 NL speakers and 5,000 background (non-NL) speakers, is used for PLDA model training. There are three utterances from each NL speaker, the mean of the three i-vectors is enrolled in NL as the utterance of the corresponding fraudster. Furthermore, similar to Khoury et al. (Khoury et al.,

2019), a normalization cohort of 4,000 augmented i-vectors is generated by applying at random a weighted sum between non-NL speaker i-vectors and NL speaker i-vectors, limiting the maximum weight for NL speakers to 20%. The Development set, consisting of one utterance from each of 3,631 NL speakers and 5,000 non-NL speakers, is used to verify the baseline approach of PLDA with AS-Norm, including the tuning of adaptive length K in AS-Norm for baseline EER computation on the Test set.

A stratified 50/50 random split of MCE Test set, consisting of one utterance from each of 3,631 NL speakers and 12,386 non-NL speakers, produces equal-sized Validation set and Evaluation set. The Validation set is used for tuning of hyper-parameters including LSH depth L, adaptive length K in NL-Norm Eq.(8), and K_t and K_e in Eqs.(10) and (11). Evaluation set is reserved for holdout testing only. Unless specified otherwise, all results presented in this paper are obtained using the Evaluation set. The motivation for generating the Validation and Evaluation sets from MCE Test set is that the 50/50 stratified split preserves the ratio of non-NL to NL speakers of the Test set, which is significantly higher than that of the Development set. It is worth noting that in real-world call center applications this ratio is much higher (Khoury et al., 2019). In addition, it is desirable that Validation and Evaluation sets have similar distributions and behaviors, whereas MCE Development set exhibits much lower EER than the Test set (Shon et al., 2019).

In contrast to MCE 2018 Challenge participating studies where the goal is to achieve the lowest EER, the aim of this work is to examine the effectiveness of the novel NL detection techniques outlined in Section 4. Therefore we adopt a minimal baseline approach: PLDA followed by AS-Norm, in order to remove potential variations introduced by nonessential steps. For example, LDA is eliminated even though it is often used prior to PLDA for dimension reduction. With this baseline, we observed NL detection EER of 1.25% and 5.66% for MCE Development and the entire Test set, respectively, which are comparable to published results in MCE 2018 Challenge (Shon et al., 2019).

PLDA implementation in the Kaldi toolkit (Povey et al., 2011) is used to generate similarity scores $s(e,t)$, $s(e,c_i)$ and $s(t,c_i)$. After score normalization, for each test utterance t the highest score among its normalized scores with NL utterances is used for the overall EER calculation (Singer and Reynolds, 2004). In our work, LSH is applied prior to PLDA to reduce the number of similarity score computations. We use the NearPy Python framework[1] as the LSH implementation with random hyperplane-based hash functions.

All computation times (i.e., detection latency) indicated in this paper were measured on an Intel Core i7-7700HQ 2.8GHz CPU with 16GB RAM.

6 Results and Discussion

6.1 NL-Norm

Applying Eq.(8) of NL-Norm, the resulting EER of NL detection is 5.57%, reduced from 5.69% for AS-Norm. To verify the stability and consistency of this result, a cross validation is performed by repeating 10 times the stratified random 50/50 split of MCE Test set into validation and evaluation sets, the outcome is shown in Table 2, where an average of 0.11% reduction in EER absolute value is observed.

Table 2: NL-Norm vs AS-Norm EER

		AS-Norm	**NL-Norm**
EER (Evaluation Set)		5.69%	5.57%
EER (Cross Validation)	mean	5.64%	5.53%
	std	0.26%	0.28%

It is worth noting that because NL-Norm considers the scores between the normalization cohort and the entire NL cohort as a single distribution, it can diminish the distinctions among individual NL speakers. Therefore we postulate that NL-Norm may not improve multi-target speaker identification (i.e. Top-1 identification). This is beyond the current NL detection study and remains to be examined in the future.

6.2 Adaptive Length K_e and K_t

Comparing AS-Norm of Eq.(10) with Eq.(5), with the additional tuning parameter, a lower EER can be reached when separate adaptive lengths K_e and K_t are employed. Table 3 shows the tuning progress on the Validation Set, where the first row represents AS-Norm that requires $K_e = K_t$, and the second row represents the best result found with $K_e = 1600$ and $K_t = 600$. In addition to the gain in accuracy, more importantly, this approach offers the flexibility of selecting a low K_t value that yields a close-to-optimal EER, as demonstrated by the last row of Table 3, where $K_t = 200$. A low K_t,

[1] https://github.com/pixelogik/NearPy

combined with LSH, enables a significant reduction in the number of score computations between the test utterance t and the normalization cohort, which in turn increases the NL detection speed. The detail is presented in the next section.

Table 3: AS-Norm with Different Adaptive Lengths

K_e	K_t	EER (%)
300	300	5.69
1600	600	5.61
3800	200	5.62

6.3 The utility of LSH

Consider a test vector, t, which must be compared against a NL of size $E = 3,631$. As Table 4 shows, the fastest distance algorithm for comparison is cosine distance, which takes 4ms to compare t against all of the NL entries. However, its speed is achieved at the cost of accuracy: the cosine distance yields an EER of 7.40%. PLDA improves on the EER but at the expense of an increased latency. LSH followed by PLDA allows us to not only derive an EER similar to that of PLDA only, but it also does so at a fraction of time: 28ms compared to 864ms for PLDA. For the test vector t, LSH search is first conducted to find L members of NL that are most similar to t, then PLDA scores between t and these L utterances are computed, with the highest score selected for EER calculation.

Table 4: Accuracy and Latency

Algorithm	**EER (%)**	**Time (ms)**
Cosine distance	7.40	4
PLDA	6.49	864
LSH + PLDA (L=30)	6.46	28

When the LSH depth L increases, it is expected the resulting EER will approach the EER exhibited by PLDA if a greedy score calculation strategy is used across the entire NL. Such an expectation is demonstrated empirically in Figure 2, where the red line represents the EER obtained from PLDA (without score normalization). For the NL dataset with size $E = 3,631$, an LSH depth $L = 30$ is already sufficient to match the EER of PLDA only.

It should be noted that even though EER $= 6.46\%$ at $L = 30$, which is slightly better than the EER of 6.49% without LSH, it is not an indication that LSH can help improve NL detection accuracy. The reason for the occasionally lower EER when performing LSH first is that for a test utterance not spoken by any NL speaker, LSH may fail to find the NL member that would have produced the highest PLDA score, thus eliminating a false-match instance from the EER evaluation.

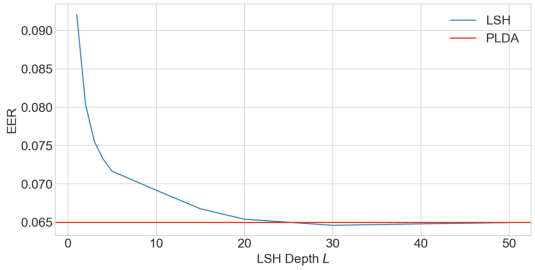

Figure 2: EER vs.LSH depth L without score normalization

As seen in Fig. 2, this effect quickly disappears with increasing LSH depth L. The results in Fig. 2 are obtained without score normalization. Score normalization improves EER, we thus examine the effect of combining LSH, PLDA and score normalization next.

6.4 Putting it together: LSH and NL-Norm

Algorithm 1 presents our use of LSH and NL-Norm. Lines 1-12 establish the normalization over the entire NL cohort as discussed in Section 4.1; this step is done only once, during initialization. Each enrolled speaker's utterances are scored against all of the normalization cohorts and the top K_e matches for that speaker and the normalization cohort are saved (line 10). At the end of the loop on line 11, $S_{NL}^{(K_e)}$ is populated and will be used to compute the NL-Norm later. As part of populating $S_{NL}^{(K_e)}$, it is possible that the eventual top-ranked NL speaker which induces the highest normalized PLDA score is not among the top K_e NL members identified by the LSH search, resulting in a decrease in the NL detection accuracy. This deficiency can be mitigated by expanding LSH search depth K_e. Line 12 invokes the NL-Norm computation function that returns true if a match is found.

For a given target vector t, we first find the top-L nearest neighbours of t from the NL $\mathcal{E}$ (line 14). To apply normalization, PLDA scores between t and utterances in the normalization cohort $\mathcal{C}$ are also needed; LSH can be utilitized once more to find K_t utterances in $\mathcal{C}$ that are most similar to t (line 15). By the end of the loop on line 19, both the sets $S_t^{(K_t)}$ and $S_{NL}^{(K_e)}$ are available, the latter populated during initialization as described in the preceding paragraph. Finally, lines 21-25 computes the normalized PLDA score between t and each NL list member, saving the results in a list (Line 25) that is checked against the threshold to determine an NL match. With Algorithm 1 the total

number of PLDA evaluations are reduced from $O(E+N)$ without LSH to $O(L+K_t)$ with LSH, where $L \ll E$ and $K_t \ll N$.

Algorithm 1: LSH and NL-Norm

Data: t: Test vector to be matched in $\mathcal{E}$
Data: T: Threshold for NL detection
Data: L, K_t, K_e: LSH depth and adaptive lengths, selected using Validation set (Sec. 5)
Data: $\mathcal{C}$: The normalization cohort
Data: $\mathcal{E}$: The negative list

$initialized \leftarrow False$;
if *(!initialized)* **then**
  $initialized \leftarrow True$;
  $S_{NL}^{(K_e)} \leftarrow \emptyset$;
  for $(e \in \mathcal{E})$ **do**
    $S_e \leftarrow \emptyset$;
    for $(c \in \mathcal{C})$ **do**
      $S_e \leftarrow S_e \cup score(e,c)$;
    end
    $S_{NL}^{(K_e)} \leftarrow S_{NL}^{(K_e)} \cup top\_n(S_e, K_e)$ // Eq.9
  end
$return\ NL\_norm(\mathcal{E}, \mathcal{C}, S_{NL}^{(K_e)}, t, L, K_t, T)$
function *boolean NL_norm($\mathcal{E}$: list, $\mathcal{C}$: list, $S_{NL}^{(K_e)}$: list, t: vector, L: int, K_t: int, T: float)*:
  $t\_\mathcal{E}\_set \leftarrow LSH\_lookup(t, \mathcal{E}, L)$;
  $t\_\mathcal{C}\_set \leftarrow LSH\_lookup(t, C, K_t)$;
  $S_t^{(K_t)} \leftarrow \emptyset$;
  for $(q \in t\_\mathcal{C}\_set)$ **do**
    $S_t^{(K_t)} \leftarrow S_t^{(K_t)} \cup score(t,q)$ // Eq.4
  end
  $result\_set \leftarrow \emptyset$;
  for $(e \in t\_\mathcal{E}\_set)$ **do**
    $s \leftarrow score(e,t)$;
    $s_{nl\_norm} \leftarrow \frac{1}{2}\left(\frac{s-\mu(S_{NL}^{(K_e)})}{\sigma(S_{NL}^{(K_e)})} + \frac{s-\mu(S_t^{(K_t)})}{\sigma(S_t^{(K_t)})}\right)$ // Eq.11
    $result\_set \leftarrow results\_set \cup \{s_{nl\_norm}\}$;
  end
  $return\ max(result\_set) > T ?\ True\ \ False$;

Table 5 lists the results obtained using various approaches, including the NL detection time per test utterance along with parameters L, K_e and K_t which, as described in Section 5, are selected via a grid search for the lowest EER on the Validation set, then applied to the Evaluation set . As demonstrated in Section 4.2, if multiple $\{K_e, K_t\}$ pairs yield near-lowest EERs for the Validation set, then the one with a low K_t is selected to take full advantage of LSH.

The first two rows in Table 5 are results of the baseline model, with and without score normalization. By applying LSH without score normalization, the NL detection time per test utterance is reduced from 864ms to 28ms, with little change in EER. When adopting Eq.(11) for score normalization, EER is lowered to 5.48% by taking advantage of NL-Norm and allowing different adaptive lengths K_t and K_e, at the same time the NL detection time is shortened significantly from 1975ms to 102ms, a beneficiary of LSH search. As noted previously, these results are obtained using the 600-dimension i-vector as input to PLDA model directly. A dimension reduction step such as LDA can be inserted before PLDA to reduce the inference time further for all approaches listed in Table 5, with or without LSH search. Nonetheless the use of LSH already make real-time NL detection feasible for applications such as call center services.

Table 5: EER and NL Detection Time (per utterance)

Method	EER (%)	L	K_e	K_t	Time (ms)
PLDA, no score norm	6.49	-	-	-	864
AS-Norm (Eq.(5))	5.69	-	300	300	1975
NL-Norm (Eq.(8))	5.57	-	500	500	1981
AS-Norm (Eq.(10))	5.52	-	3800	200	2051
NL-Norm (Eq.(11))	5.57	-	400	500	1976
LSH + PLDA, no score norm	6.46	30	-	-	28
LSH + AS-Norm (Eq.(10))	5.66	50	2000	250	114
LSH + NL-Norm (Eq.(11))	5.48	50	3700	200	102

7 Conclusion

Negative list detection is an important application for fraud detection in various industries such as call center services. This work explored novel techniques specifically devised for NL detection, with the aim of improving both accuracy and speed. NL-Norm considers similarity scores between the normalization cohort and all enrolled NL speakers as a single distribution, which helps calibrate test scores and select a consistent threshold over the entire NL. LSH is applied to find NL speakers as well as utterances in the normalization cohort that are most similar to a test utterance, so that PLDA scoring is performed only on a small subsets of utterances, which significantly lowers the computation cost and latency of the NL detection. The effectiveness of LSH is further amplified when using different adaptive lengths for Z-Norm and T-Norm terms in AS-Norm and NL-Norm, so that evaluating a relatively small number of similarity scores between a test utterance and the normalization cohort is sufficient to reach optimal accuracy.

While the experiments are conducted on i-vectors, none of the techniques proposed in this paper is specific to i-vector, therefore we expect these approaches can be applied to x-vectors as well, which will be the subject of a future study, along with more direct comparisons with results from other NL detection methods.

References

Roland Auckenthaler, Michael Carey, and Harvey Lloyd-Thomas. 2000. Score normalization for text-independent speaker verification systems. *Digital Signal Processing*, 10(1):42–54.

Moses S. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *Proc. of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, STOC '02, page 380–388. ACM.

Sandro Cumani, Pier Batzu, Daniele Colibro, Claudio Vair, Pietro Laface, and Vasileios Vasilakakis. 2011. Comparison of speaker recognition approaches for real applications. In *Proc. of the Annual Conf. of the Intl. Speech Communication Association, INTERSPEECH*, pages 2365–2368.

Najim Dehak, Patrick J. Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. 2011. Front-end factor analysis for speaker verification. *IEEE Trans. on Audio, Speech, and Language Processing*, 19(4):788–798.

Roberto Font. 2019. A denoising autoencoder for speaker recognition. results on the mce 2018 challenge. In *ICASSP 2019 - 2019 IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6016–6020.

J. Fortuna, P. Sivakumaran, A. Ariyaeeinia, and A. Malegaonkar. 2005. Open-set speaker identification using adapted Gaussian mixture models. In *Proc. Interspeech 2005*, pages 1997–2000.

J. Fortuna, P. Sivakumaran, A. M. Ariyaeeinia, and A. Malegaonkar. 2004. Relative effectiveness of score normalisation methods in open-set speaker identification. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)*, pages 369–376.

Craig S. Greenberg, Lisa P. Mason, Seyed Omid Sadjadi, and Douglas A. Reynolds. 2020. Two decades of speaker recognition evaluation at the national institute of standards and technology. *Computer Speech & Language*, 60:101032.

Nancie Gunson, Diarmid Marshall, and Mervyn Jack. 2015. Effective speaker spotting for watch-list detection of fraudsters in telephone banking. *IET Biometrics*, 4(2):127–136.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proc. of the 30th Annual ACM Symposium on Theory of Computing*, STOC '98, page 604–613, New York, NY, USA. ACM.

Woojay Jeon and Yan-Ming Cheng. 2012. Efficient speaker search over large populations using kernelized locality-sensitive hashing. In *2012 IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4261–4264.

Zahi Karam, William Campbell, and Najim Dehak. 2011. Towards reduced false-alarms using cohorts. In *Acoustics, Speech, and Signal Processing, 1988. ICASSP-88., 1988 Intl. Conf. on*, pages 4512–4515.

Patrick Kenny. 2010. Bayesian Speaker Verification with Heavy-Tailed Priors. In *Proc. The Speaker and Language Recognition Workshop (Odyssey 2010)*, page paper 14.

Elie Khoury, Khaled Lakhdhar, Andrew Vaughan, Ganesh Sivaraman, and Parav Nagarsheth. 2019. Pindrop Labs' Submission to the First Multi-Target Speaker Detection and Identification Challenge. In *Proc. Interspeech 2019*, pages 1502–1505.

Ryan Leary and Walter Andrews. 2014. Random projections for large-scale speaker search. In *Proc. Interspeech 2014*, pages 66–70.

K. Li and J. Porter. 1988. Normalizations and selection of speech segments for speaker recognition scoring. In *ICASSP-88., Intl. Conf. on Acoustics, Speech, and Signal Processing*, pages 595–598.

Jeff Ma, Jan Silovsky, Man-hung Siu, and Owen Kimball. 2015. Large-scale speaker search using plda on mismatched conditions. In *2015 IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1846–1850.

Amit Malegaonkar and Aladdin Ariyaeeinia. 2011. Performance evaluation in open-set speaker identification. In *Biometrics and ID Management*, pages 106–112, Berlin, Heidelberg. Springer Berlin Heidelberg.

Pavel Matejka, Ondrej Glembek, Fabio Castaldo, Md Jahangir Alam, Oldrich Plchot, Patrick Kenny, Lukas Burget, and Jan Cernocký. 2011. Full-covariance ubm and heavy-tailed plda in i-vector speaker verification. In *ICASSP, IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*, pages 4828–4831.

Pavel Matějka, Ondřej Novotný, Oldřich Plchot, Lukáš Burget, Mireia Diez Sánchez, and Jan Černocký. 2017. Analysis of Score Normalization in Multilingual Speaker Recognition. In *Proc. Interspeech 2017*, pages 1567–1571.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukáš Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlíček, Yanmin Qian, Petr Schwarz, Jan Silovský, Georg Stemmer, and Karel Vesel. 2011. The Kaldi speech recognition toolkit. *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*.

Simon J.D. Prince and James H. Elder. 2007. Probabilistic linear discriminant analysis for inferences about identity. In *2007 IEEE 11th Intl. Conf. on Computer Vision*, pages 1–8.

Douglas A. Reynolds, Thomas F. Quatieri, and Robert B. Dunn. 2000. Speaker verification using adapted gaussian mixture models. *Digital Signal Processing*, 10(1):19–41.

Seyed Omid Sadjadi, Craig Greenberg, Elliot Singer, Douglas Reynolds, Lisa Mason, and Jaime Hernandez-Cordero. 2020. The 2019 NIST Speaker Recognition Evaluation CTS Challenge. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)*, pages 266–272.

Ludwig Schmidt, Matthew Sharifi, and Ignacio Lopez Moreno. 2014. Large-scale speaker identification. In *2014 IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1650–1654.

Suwon Shon, Najim Dehak, Douglas Reynolds, and James Glass. 2019. MCE 2018: The 1st Multi-Target Speaker Detection and Identification Challenge Evaluation. In *Proc. Interspeech 2019*, pages 356–360.

Suwon Shon, Younggun Lee, and Taesu Kim. 2018. Large-scale speaker retrieval on random speaker variability subspace. *arXiv preprint arXiv:1811.10812*.

Elliot Singer and Douglas A. Reynolds. 2004. Analysis of multitarget detection for speaker and language recognition. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)*, pages 301–308.

David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. 2018. X-vectors: Robust dnn embeddings for speaker recognition. In *2018 IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5329–5333.

Kevin Wilkinghoff. 2020. On Open-Set Speaker Identification with I-Vectors. In *Proc. The Speaker and Language Recognition Workshop*, pages 408–414.

Yaniv Zigel and Moshe Wasserblat. 2006. How to deal with multiple-targets in speaker identification systems? In *2006 IEEE Odyssey - The Speaker and Language Recognition Workshop*, pages 1–7.