### BLACKLISTED SPEAKER IDENTIFICATION USING TRIPLET NEURAL NETWORKS

Kin Wai Cheuk, Balamurali B T, Gemma Roig, Dorien Herremans

Singapore University of Technology and Design

#### **ABSTRACT**

We present a novel approach to the speaker recognition problem using Triplet Neural Networks. Currently, the three most popular methods includes Gaussian Mixture Model with Universal Background Model (GMM-UBM), Joint Factor Analysis (JFA), and i-vectors approach. Among them, the i-vectors approach is considered to be the state-of-the-art recognition method, as they often outperform both GMM-UBM and JFA. In this paper, we report that a Triplet Neural Network (TNN), an enhanced version of Siamese networks, in combination with different classifiers such as SVM, can outperform the state-of-the-art i-vector method. For this first Multi-target speaker detection and identification challenge at ICASP, our novel TNN approach reached an error rate (ERR) of 0.84% when combined with K-nearest-neighbor (KNN) (baseline model 2.00%) for task 1. For task 2 of this challenge, TNN with a support vector machine (SVM) is used to obtain a further improvement, improving from 444 confusion errors in the baseline model to only 359 when trying to identify the blacklisted speaker's identity.

*Index Terms*— TNN, KNN, SVM, *i*-vector, speaker classification, speaker identification, triplet networks

## 1. INTRODUCTION

This paper offers a novel approach based on Tripal Neural Networks to the speaker identification problem. We test our results on the Multi-target Challenge (MCE2018), a competition that aims to test the state-of-the-art algorithms that can recognize blacklisted speakers by analyzing audio recordings [1]. The dataset of the MCE2018 consists of *i*-vectors [2] of real-world telephone conversations by customers and agents from a call-center. The resulting problem is a multi-target speaker detection challenge. In this paper we tackle two tasks. Task 1 consists of classifying the *i*-vectors between blacklisted and non-blacklisted speakers while task 2 is a multi-target classification problem for the 3631 unique speakers in the blacklist.

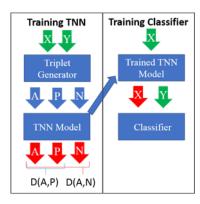
Current state-of-the-art algorithms for solving this problem include Gaussian Mixture Model with Universal Background Model (GMM-UBM) [3], Joint Factor Analysis (JFA) [4], and *i*-vectors [5, 6]. The latter have recently shown to be the method of choice, as they can outperform both GMM-UBM and JFA. In the next sections, we will describe a novel approach that includes the use of a Triplet Neural Network [7] to solve Task 1 and Task 2 of the MCE2018 competition.

### 2. SYSTEM DESCRIPTION

# 2.1. Triplet Neural Networks

In order to tackle the two classification tasks, blacklisted speaker detection and speaker identification. We first create a new representation using a Triplet Neural Network, which is then used as input in the respective classification algorithm (see Fig. 1).

A Triplet Neural Network (TNN) is a special type of networks, that are able to create new, multi-dimensional representations of input data [7]. The networks is able to create this representation, by learning a new, non-linear distance metric, that is trained to maximize the distance between different elements, and minimize the distance between similar elements. We therefore train the network on a triplets consisting of two pairs: an anchor-positive pair (i.e., i-vectors in the same class, referred as A-P in the equation (1)), and anchor-negative pairs (i-vector that in different classes, referred as A-N in the Equation 1).



**Fig. 1.** Training TNN and a classifier. The classifier could be either cosine similarity, SVM or KNN.

In order to train the Triplet network, we need to preproces our data. We first create a set of triplet data namely A - P, and A - N (indicated by blue arrow in Fig. 1), whereby A is

the anchor i-vector, P is a positive example i-vector, meaning it belongs to the same class as A and N is a negative example i-vector, meaning it belongs to a different class of A. The dataset provides us with a training set consists of i-vectors (X) and speaker labels (Y), which we can use to generate the above described triplets (represented by the green arrow in Fig. 1).

Using the generated triplet data, the TNN network can be trained, thus outputting a new representation for the given i-vectors. We use a single layer network with 600 nodes for the TNN. During training, the Euclidean distance D between the output of the network after feeding A and P, and between the output of A and A pairs are calculated. The training objective of the TNN is to minimize the loss is shown in Equation 1. It is important to note that the TNN shares the same network (with exactly the same weights) for each of the three inputs. The training objective minimizes:

$$L(A, P, N) = max(D(A, P) - D(A, N) + \alpha)$$
 (1)

Whereby L is the loss and  $\alpha$  is a factor called margin which determines how far apart the clusters be. In order to minimize L, D(A,P) should be minimized and D(A,N) should be maximized [7].

The resulting network will output a new representation when an i-vector is fed as input. This new vector will be used in for our classification tasks, as explained in the next section.

## 3. HYBRID SYSTEM ARCHITECTURE

Figure 1 displays the hybrid architecture that we use for the two classification tasks. The same TNN architecture is used, however, the weights are different between the two tasks, as we use different input labels to train (i.e. blacklisted/non-blacklisted versus speaker id).

### 3.1. Learning new multi-dimensional representations

The neural network that is embedded in the TNN is composed of fully connected layer consisting 600 neurons with Relu activation. Furthermore, an Adam optimizer with learning rate  $10^{-5}$  is used to minimize the TNN loss function described in Equation 1. The output of the TNN network is a new multi-dimensional representation of our data. Then we feed this new representation to train a classifier algorithm. After training, the classifier would be able to predict the class of a ivector transformed by the TNN. In this paper, we have compared a support vector machines classifier (SVM), k-nearest neighbors (KNN), and cosine similarity scoring.

### 3.2. Classification algorithms

Firstly, we implemented SVM with Radial basis function (RBF) as the kernel. Grid-search [8] was used to determine the best performing parameters, which resulted in a penalty

factor C=1 and gamma set to 1/600. For KNN, the number of neighbors is set to 3 for Task 1 and set to 1 for Task 2. The cosine similarity model is just the same as the baseline model.

## 3.2.1. Task 1: blacklisted speaker detection

The aim of this task is to identify whether a given i-vector belongs to a blacklisted speaker or not. It is a binary classification problem, of which the labels for Y are either 1 (blacklisted) or 0 (background). In order to solve this task, our triplet generator will provide us with a set of triplets based on our two classes. This will allows the TNN to learn a new representation of our i-vectors that maximizes the distance between blacklisted and non-blacklisted speakers.

Given that the dataset consists of 3631 unique black-listed speakers (3 samples for each speaker) and 5000 non-blacklisted speakers (more than 4 samples each), the number of possible A-P-N triplets that we can generate is huge. Therefore, in order to avoid memory issues, we randomly sample some of the possible combinations and train the network for a number of epochs, after which point, another set of A-P-N triplets are sampled and training continues. The network continues training until the loss approaches zero.

For Task 1, the number of sampled triplets for each batch is 96,000, these are trained for 30 epochs. After generating 4 different sets of samples (i.e., 120 epochs in total), the loss is nearly zero, and the weights that give rise to the best result (i.e. lowest error rate ERR when using the baseline cosine similarity classifier without M-norm) are chosen for the TNN model.

In addition to the cosine similarity classifier, performance of two other classifiers such as K-nearest neighbour (KNN), and SVM are investigated. These three models are named as TNN-cosine, TNN-KNN, and TNN-SVM respectively. The results are discussed in the next section.

### 3.2.2. Task 2: speaker identification

Assuming that an unknown i-vector belongs to a blacklisted speaker, the aim of Task 2 is to find out the identity of the speaker from the dataset of 3631 blacklisted speakers. This is a non-trivial task, as there are only three samples provided per speaker.

In order to tackle this speaker identification problem, we first re-train the TNN to generate new representations of our *i*-vector inputs, based the distances *between the different black-listed speakers*, as opposed to the distances between blacklisted and non-blacklisted speakers (as in Task 1). This is done by re-training the exact same TNN networks (same architecture, different weights), on a new set of triplets.

For each batch, we sampled 1,000,000 combinations during training. After each 5 epochs, a new batch was sampled to continue the TNN training (similar to the procedure for Task

1), until the loss was near to zero. Finally, Task 2 is predicted using TNN-cosine, TNN-KNN, and TNN-SVM classification models.

#### 4. RESULTS

The performance of our hybrid TNN-based classifiers are shown in Table 1. For blacklisted speaker detection (Task 1), the TNN-KNN (when the n-neighbors is 3) was found to be the best performing candidate among the three models and was chosen to be the primary model. The final error rate ERR is 0.84%.

	Task 1 (ERR)	Task 2 (Confusions)
TNN-cosine	1.22%	380
TNN-KNN TNN-SVM	<b>0.84%</b> 1.44%	428 <b>358</b>
baseline	2.00%	444

**Table 1**. Models' performance of both task

For Task 2, speaker identification, we found that the triplet neural network with support vector machine classifier (TNN-SVM) outperforms the TNN with cosine distance-based classifier ((358 versus 380 confusion errors). When compared to TNN with K-nearest neighbor, the TNN-SVM is clearly the best performing algorithm. Therefore, TNN-SVM is the primary model for Task 2.

Since different hybrid models perform differently on the two tasks, we therefore choose the following models for submission. The difference between Primary and Contrastive 1 submission on the weights of the TNN. While in Contrastive2, we use cosine similarity scoring instead of KNN for task 1, and again, we choose another set of weights for the TNN model in task 2. All of the models are trained with only training set, and validate on the development set only.

	Task 1 (ERR)	Task 2 (Confusions)
Primary	TNN-KNN (0.84%)	TNN-SVM (358)
Contrastive1	TNN-KNN (0.84%)	TNN-SVM (358)
Contrastive2	TNN-cosine (1.22%)	TNN-SVM (359)

#### 5. ACKNOWLEDGEMENTS

This work is supported by the SUTD SRG ISTD 2017 129 grant and the SMART-MIT Innovators grant ING1611118-ICT.

#### 6. REFERENCES

- [1] Suwon Shon, Najim Dehak, Douglas Reynolds, and James Glass, "Mce 2018: The 1st multi-target speaker detection and identification challenge evaluation (mce) plan, dataset and baseline system," *ArXiv e-prints arXiv:1807.06663*, 2018.
- [2] Suwon Shon, Najim Dehak, Douglas Reynolds, and James Glass, "Mce 2018: The 1st multi-target speaker detection and identification challenge evaluation (mce) plan, dataset and baseline system," *arXiv preprint arXiv:1807.06663*, 2018.
- [3] Taufiq Hasan and John HL Hansen, "A study on universal background model training in speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 1890–1899, 2011.
- [4] Patrick Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," *CRIM*, *Montreal*,(*Report*) *CRIM-06/08-13*, vol. 14, pp. 28–29, 2005.
- [5] Frank K Soong, Aaron E Rosenberg, Bling-Hwang Juang, and Lawrence R Rabiner, "Report: A vector quantization approach to speaker recognition," *AT&T technical journal*, vol. 66, no. 2, pp. 14–26, 1987.
- [6] Daniel Garcia-Romero and Carol Y Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in Twelfth Annual Conference of the International Speech Communication Association, 2011.
- [7] Elad Hoffer and Nir Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*. Springer, 2015, pp. 84–92.
- [8] PM Lerman, "Fitting segmented regression models by grid search," *Applied Statistics*, pp. 77–84, 1980.