

# Experiments report

PS: 红色表示需要修改的地方

## Introduction

The goal of this work is to improve SRE performance in PLDA score using Variational Autoencoder and triplet loss.

Probabilistic linear discriminant analysis (PLDA) is the defector standard for backends in i-vector speaker recognition. But if we try to extend the PLDA in d-vector or x-vector SRE system, the performance is not as well as i-vector. One of the main causes of this problem is that i-vector follows Gaussian distribution, but d-vector and x-vector do not.

Some predecessors (Lilt...) think that there are 2 main factors determine the PLDA performance:

1. the Gauss property of embedding
2. Distinguishability between different speaker's embedding

In this work, we propose to approach this problem using stochastic gradient variational Bayes, which aims to enhance the Gauss property of d/x-vector. And we also add triplet loss into model loss function to increase the distinguishability between different speaker's embedding.

我不知道我写的英文对不对，所以用中文重新表述一下。

决定 vector 在 PLDA 的好坏有两个因素，

1. vector 是否有很强的高斯性
2. 不同说话人之间的 vector 是否有很强的区分性

这个实验的目标：

用 VAE 使 vector 高斯（但可能会降低区分性）  
因此用 triplet 使得 vector 保证有区分性

总结来说，用 VAE 使 vector 变得高斯的情况下用 triplet loss 保证区分性，从而提高 PLDA 的性能

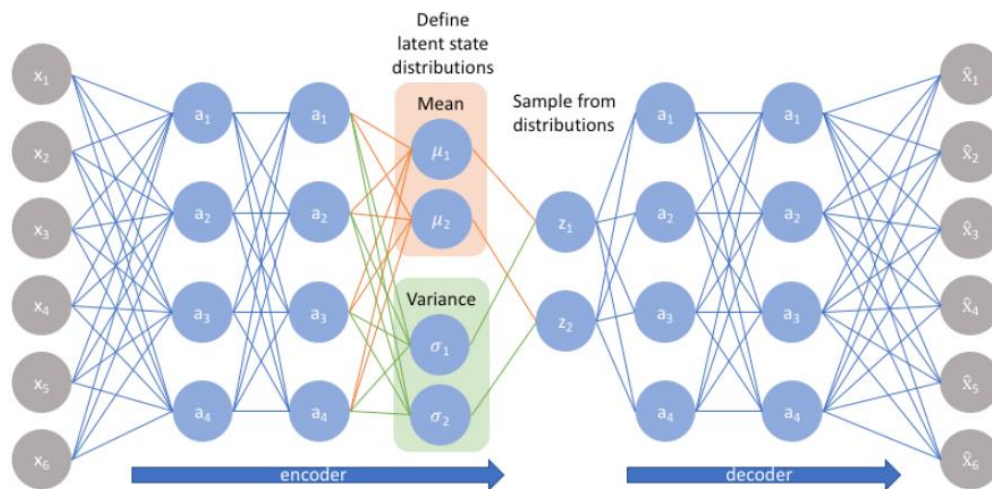
# Experiments

## Network structure

### 1. Vanilla VAE

VAE encoder: 2 hidden layers (sigmoid+tanh)

VAE decoder: 2 hidden layers (tanh+sigmoid)



We trained this model using **reparameterization trick**, this enables us to optimize of the loss function using backpropagation to jointly estimate the parameters that define the model and the approximate posteriors of the latent factors

$$\mathcal{L}(x, \hat{x}) + \sum_j KL(q_j(z|x) || p(z))$$

### 2. VAE with Triplet Loss

结构图略

Total\_loss = Loss(VAE) +  $\alpha$  \* loss(triplet loss)

## Work Implementation

1. Firstly, we use kald toolkit to extract d-vector and x-vector with 512 dimensions from utterance.
2. Then, we implemented the VAE with the TensorFlow toolkit, the inputs are d-vector and x-vector extracted from the first step.
3. When we finish the train training process, we input the original d/x-vector into network and get the output from VAE encoder.
4. At last, we use kald toolkit to calculate EER of output vector.

NOTE: I have submitted the TF-VAE code to GitHub. (<https://github.com/zyzisy/zkhk>)

## Datasets

**Train:** voxceleb\_combined\_200000

**Test:** sitw\_dev & sitw\_eval

## Baseline

	Cosine EER	PLDA EER	LDA PLDA EER
d-vector	38.39%	17.71%	9.511%
x-vector	15.67%	9.087%	3.157%

## Experiment Results:

We designed 4 groups of experiments to find the best network parameters.

1. D/X-vector -> Vanilla VAE
2. D/X-vector -> VAE with Triplet Loss
3. D/X-vector -> LDA -> Vanilla VAE
4. D/X-vector -> LDA -> VAE with Triplet Loss

## 1. D/X-vector -> Vanilla VAE

Vanilla VAE Best Result

	The Best Network structure	Cosine EER		PLDA EER		LDA PLDA EER	
		baseline	VAE	baseline	VAE	baseline	VAE
d-vector	Z dim: 600 layer unit: 1800	38.39%	<b>38.54%</b>	17.71%	<b>16.02%</b>	9.511%	<b>12.59% (75)</b>
x-vector	Z dim: 200 layer unit: 2300	15.67%	<b>13.79%</b>	9.087%	<b>5.006%</b>	3.157%	<b>4.236% (100)</b>

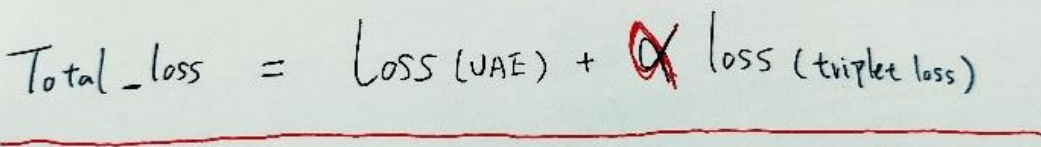
The PLDA EER of d-vector does not have an obvious enhancement. But the X-vector's result seems good.

Note:

1. Baseline means without VAE encoder.
2. The best d-vector-VAE's Z dim is 600 which is more than 512.
3. LDA PLDA EER 括号表示 LDA 的 dim

## 2. D/X-vector -> VAE with Triplet Loss

The total loss function of VAE with Triplet Loss consists of 2 parts ( VAE Loss + Triplet Loss), in this experiment, we use the Vanilla VAE best network structure we have got before to find the best hyperparameter  $\alpha$ .



Total\_loss = Loss (VAE) +  $\alpha$  loss (triplet loss)

	The Best Network structure	Cosine EER		PLDA EER		LDA PLDA EER	
		Baseline	VAE	baseline	VAE	baseline	VAE
d-vector	$\alpha$ : 150	38.39%	<b>16.1%</b>	17.71%	<b>13.67%</b>	9.511%	<b>11.17% (65)</b>
x-vector	$\alpha$ : 200	15.67%	<b>6.546 %</b>	9.087%	<b>4.467</b>	3.157%	<b>4.043% (130)</b>

### 3. D/X-vector -> LDA -> Vanilla VAE

	The Best Network structure	Cosine EER		PLDA EER		LDA PLDA EER	
		baseline	VAE	baseline	VAE	baseline	VAE
d-vector	Z dim: 80 layer unit: 1600	12.94%	<b>12.09%</b>	9.665%	<b>9.472%</b>	9.434%	<b>9.049% (70)</b>
x-vector	Z dim: 120 layer unit: 1800	5.198%	<b>4.39%</b>	3.735%	<b>3.581%</b>	3.157%	<b>3.273% (110)</b>

### 4. D/X-vector -> LDA -> VAE with Triplet Loss

	The Best Network structure	Cosine EER		PLDA EER		LDA PLDA EER	
		baseline	VAE	baseline	VAE	baseline	VAE
d-vector	Z dim: 80 layer unit: 1600	12.94%		9.665%		9.434%	
x-vector	Z dim: 120 layer unit: 1800	5.198%		3.735%		3.157%	

My basic shell script to train a model.

```
python -u main.py --epoch 15 \
    --batch_size 100 \
    --n_hidden ${n_h} \
    --learn_rate 0.00001 \
    --beta1 0.5 \
    --dataset_path ./data/voxceleb_combined_200000/dvector.npz \
    --z_dim ${z_d}
```