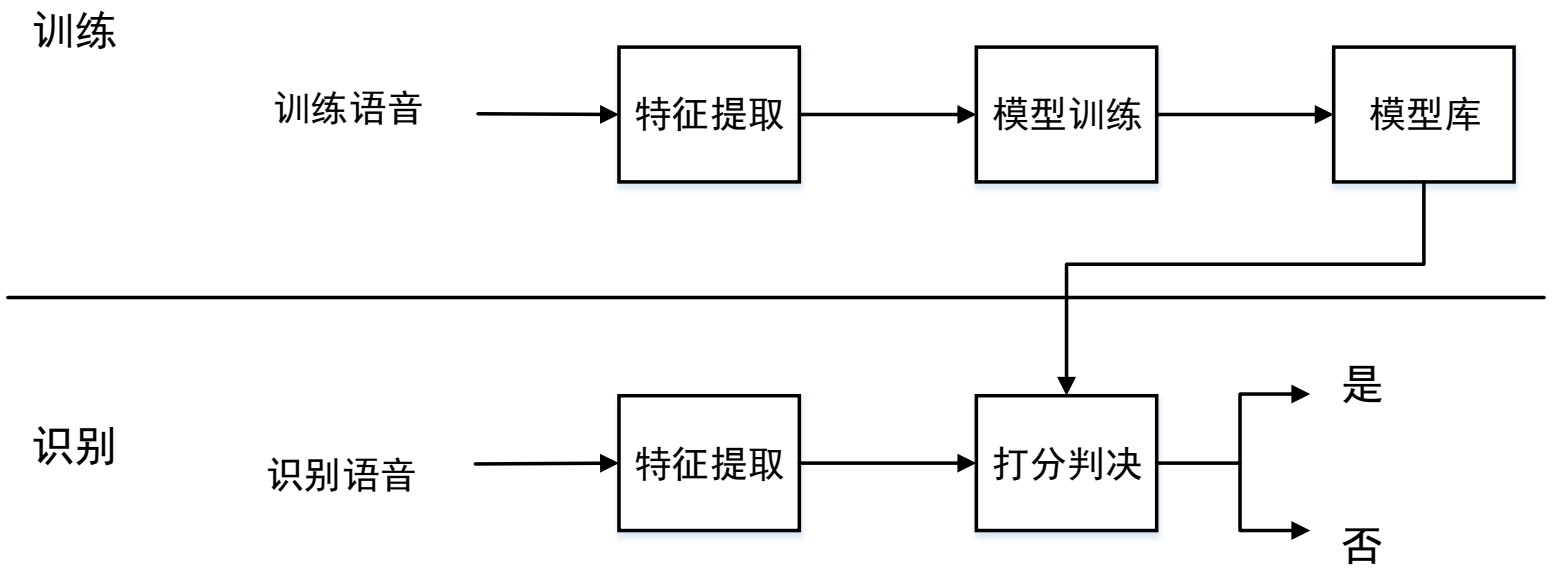
Text-Independent Speaker Verification Based on Triplet Convolutional Neural Network Embeddings

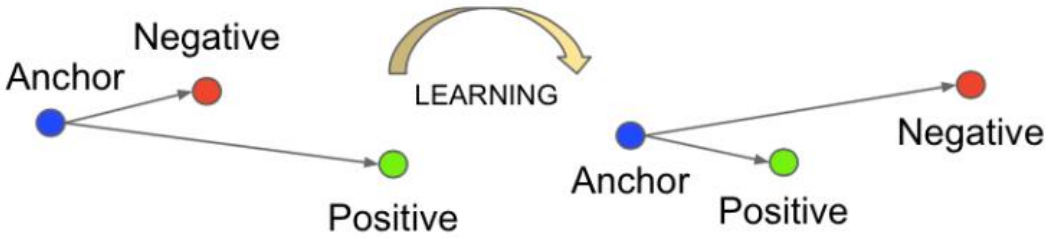
1. 声纹识别



声纹识别系统框架

声纹识别的系统框架如上图所示：整个系统分为训练阶段和识别阶段。在训练阶段，首先对训练语音进行预处理和特征提取，然后构建能代表说话人的模型，将其存放在模型库中。在识别阶段，同样需要对待识别的语音进行预处理、特征提取和模型构建，然后把得到的模型与模型库中的模型进行一对一的相似性比较，判断是否属于同一个人。

2. triplet loss



Loss:

$$\Delta_i = \|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha,$$

$$L = \sum_{i=1}^{N} \max(0, \Delta_i), (x_i^a, x_i^p, x_i^n) \in \mathcal{T}$$

Triplet loss: 首先选择一个样本Anchor，Anchor属于身份A，再选择一个样本Positive，同样属于A，然后再选择一个样本Negative，该样本不属于A，triplet loss 的目的就是使属于同一个身份的样本之间的距离尽可能地近，不同身份样本之间的距离尽可能的远，最终得到能够代表一个说话人的embedding向量。

3. 方案

该方案主要分为三个阶段。

训练阶段：

（1）. 从数据集中每次选取一个batch的triplet三元组（anchor、positive、negative）

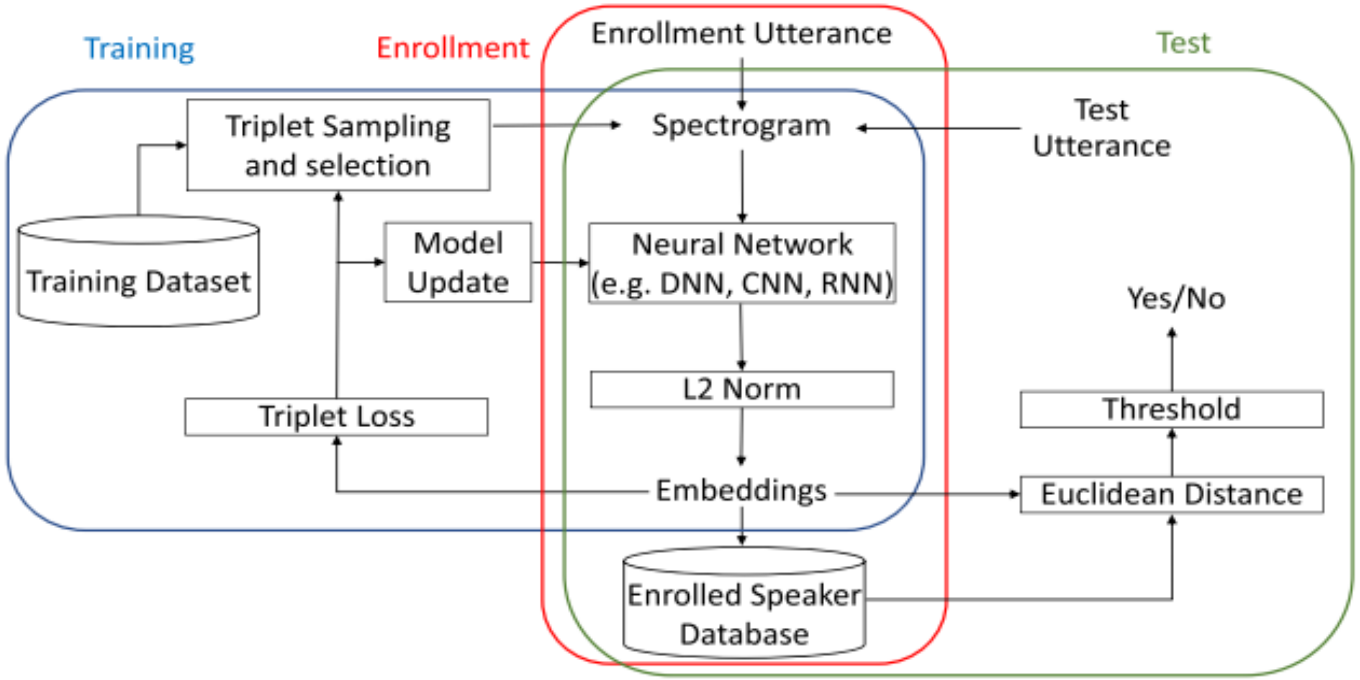
（2）. 对每条语音提取fbank特征或FFT频谱特征，每条语音得到一个二维的特征矩阵。

（3）. 把一个三元组的每个特征矩阵分别输入到网络中，得到输出结果，输出为128维向量。

（4）. 对输出的结果进行L2归一化处理。

（5）. 计算一个三元组的triplet loss，并以同样的方式计算出一个batch三元组的triplet loss

（6）.根据triplet loss计算出梯度并更新网络模型的参数。



注册阶段：

（1）. 采集说话人的语音

（2）. 对语音进行有效性检测，去除静音并分成4s的定长语音片段

（3）. 分别提取语音片段的fbank特征或者FFT变换的频谱特征。

（4）. 把提取的特征矩阵输入到训练好的网络中得到多个128维的向量

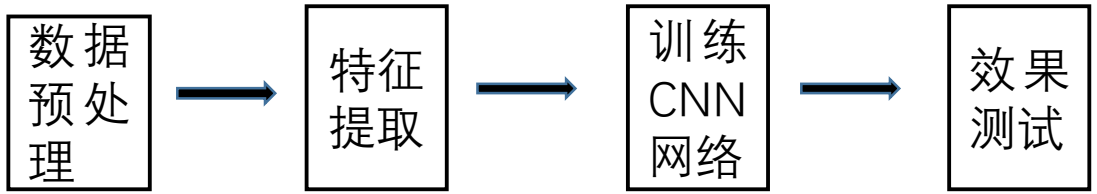
（5）. 对输出的向量分别做L2归一化，然后取平均值，再进行L2归一化，得到代表一个说话人的embedding，并存入数据库中。

测试阶段：

（1）.按照与注册阶段相同的过程，得到该语音的embedding

（2）.把该条embedding分别与数据库中的embedding做相似性比较，判断是否属于同一个人

4. 具体实现



1.数据预处理：首先对数据集中的样本进行语音有效性检测（VAD）去除静音部分，然后将语音样本进行切割，长度为4秒，滑动的step为2秒。

2.特征提取：输入特征采用两种方式：Fbank特征或FFT特征。经过预处理后的语音采样率为8K，然后进行分帧，窗口长度为0.032秒，step为0.016秒，4秒长的语音段会被分成250帧。对于Fbank特征，每帧提取120维的特征，4秒的语音形成120 * 250的特征矩阵；对于fft特征，对每帧数据做256个点的FFT，取前128个频谱分量作为特征，4秒的语音就构成128*250的特征矩阵。

3.训练CNN：构建triplet三元组，把triplet loss作为优化函数，对CNN网络进行训练。

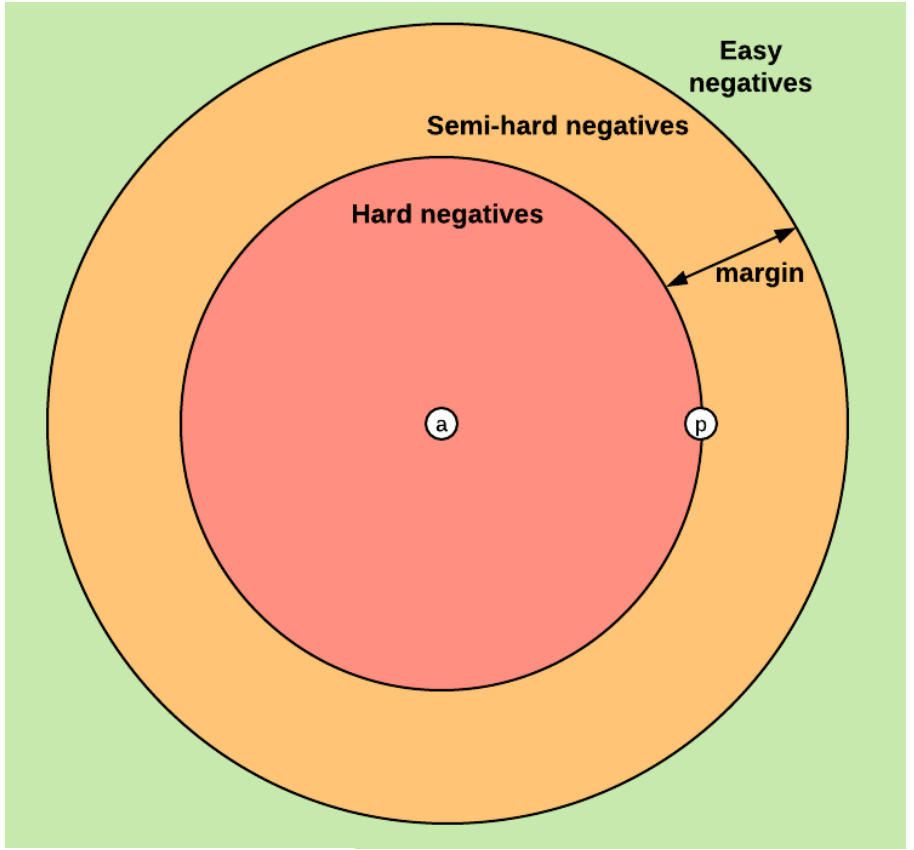
$$\Delta_i = \|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha,$$

$$L = \sum_{i=1}^{N} \max(0, \Delta_i), (x_i^a, x_i^p, x_i^n) \in \mathcal{T}$$

三元组选取

三元组的选取分为三类：

- easy triplets：可以使loss = 0的三元组，即容易分辨的三元组
- hard triplets：$d(a,n) < d(a,p)$的三元组，即一定会误识别的三元组
- semi-hard triplets：$d(a,p) < d(a,n) < d(a,p) + margin$ 的三元组，即处在模糊区域（关键区域）的三元组



离线：

可以在每轮迭代之前从所有triplet中选择semi-hard Triplet。也就是先对所有的训练集计算嵌入表达(feature)，然后只选择semi-hard triplets并以此为输入训练一次网络。因为每轮训练迭代之前都要遍历所有triplet，计算它们的嵌入，所以offline挖掘triplet效率很低

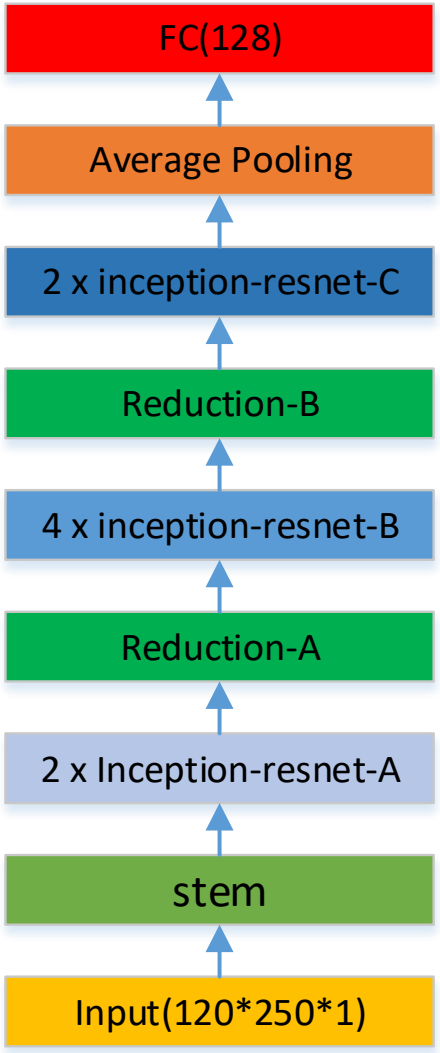
在线：

假设有B个图片（不是Triplet），也就是可以生成B个嵌入表达，那么我们最多以此生成B的三次方个Triplet，当然大多数Triplet都不符合要求，优点是只用遍历B个图片，效率高。

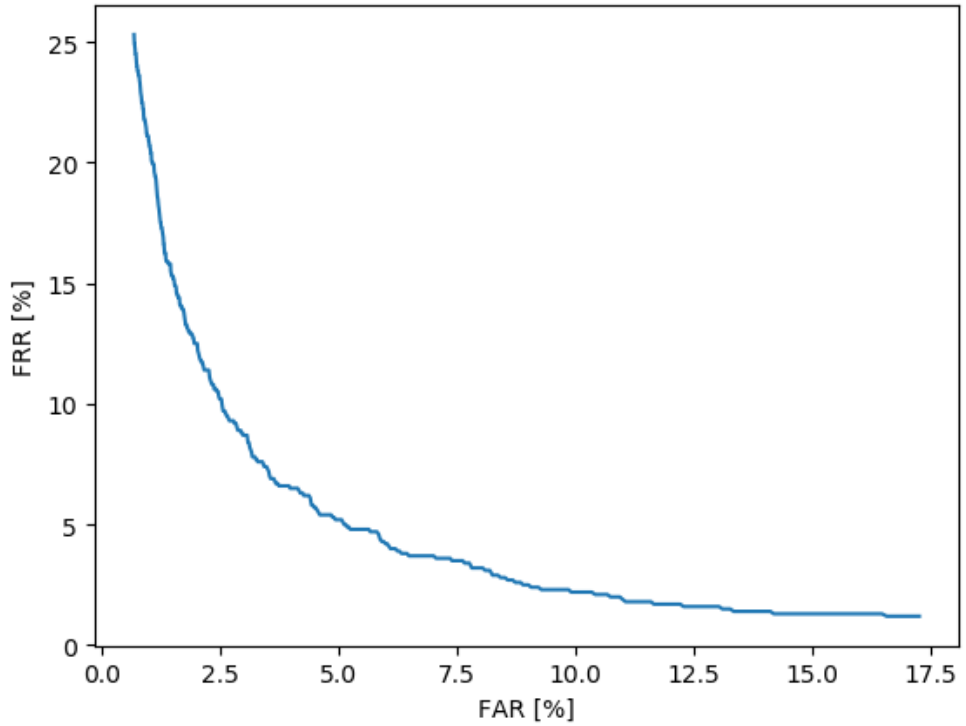
- **batch all：** 选择所有合格的Triplet，对其中的hard和semi-hard Triplet的损失取均值
  - 这里的关键在于消除easy Triplet的影响，因为easy Triplet的loss = 0，会拉低平均值
  - 合格的Triplet的数目为$PK(K-1)(PK-K)$即PK个原点，K-1个同类样本，PK-K个异类样本
- **batch hard：** 遍历所有原点(也就是banch中的所有样本），选择hardest同类样本($d(a,p)$最大的样本)，选择hardest异类样本($d(a,n)$最小的样本)
  - 一共有PK个Triplet

5. 实验及其结果

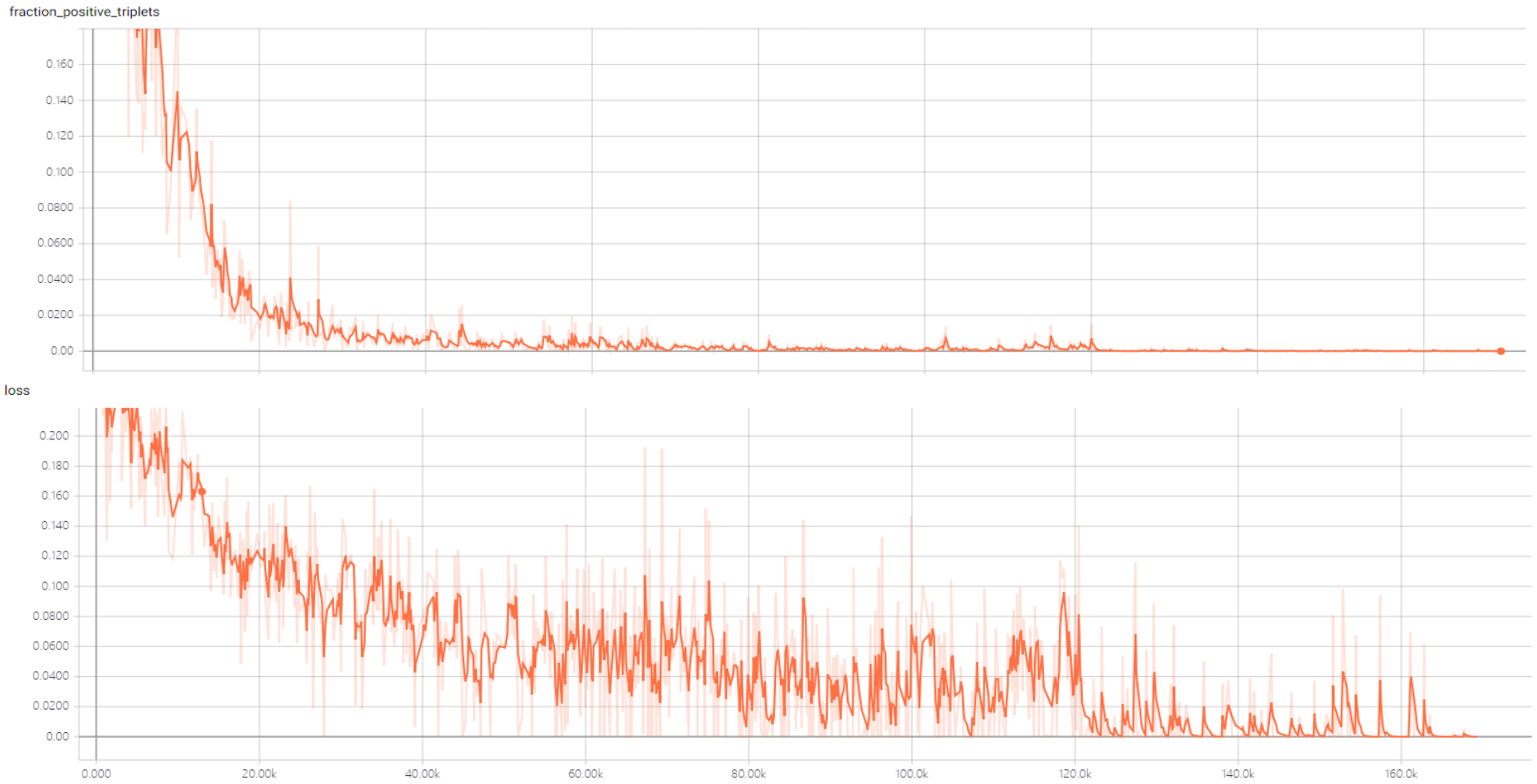
数据集采用的是NIST SRE16数据集，训练使用600个人，每个人1到3条语音，每条语音1分钟左右。测试时用200个人进行注册，每个人挑出5条长度维8秒的语音段进行测试。实验共200*5*200=200000个。



特征	EER	阈值
Fbank	5.21%	1.001



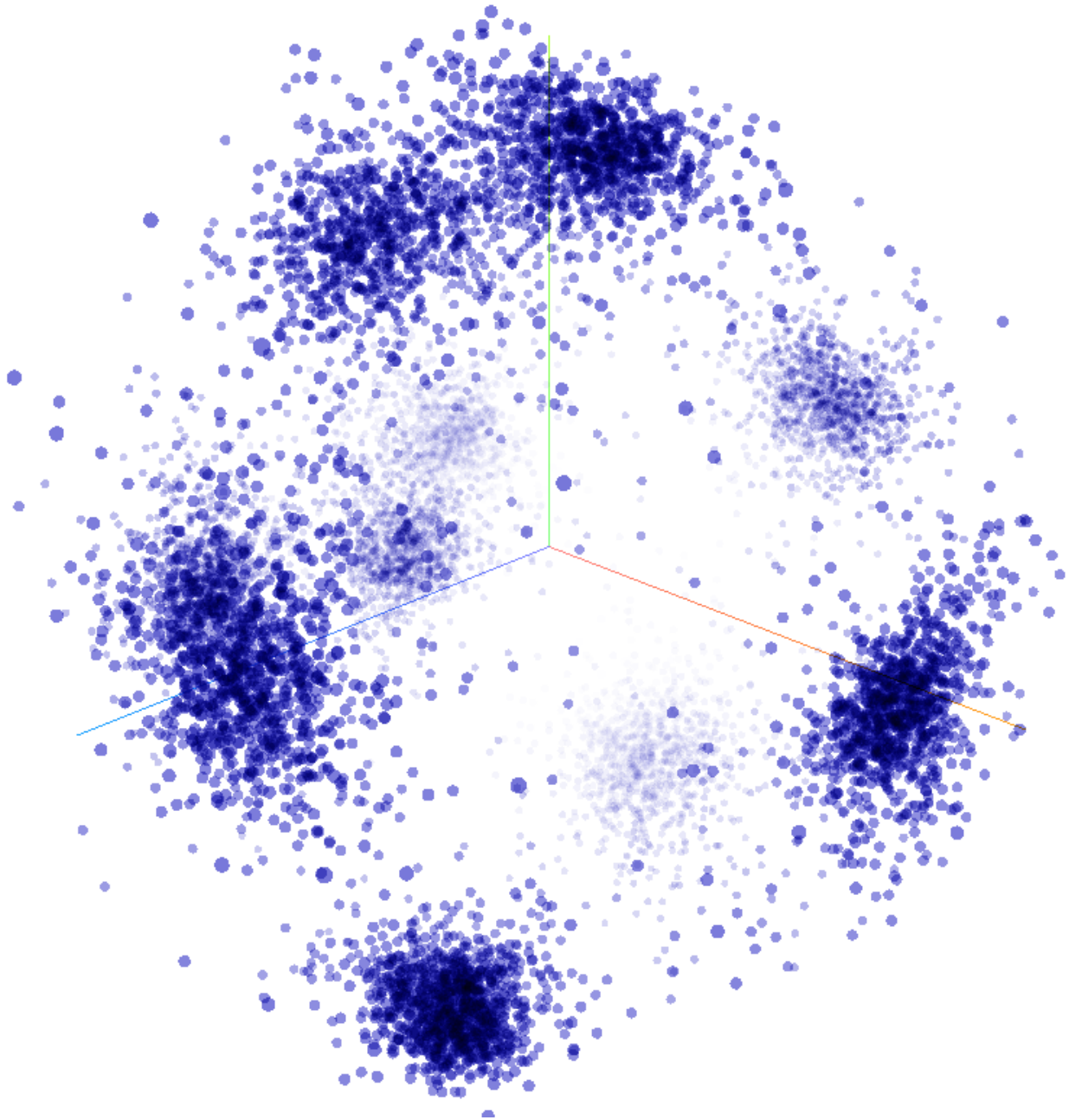
6. 优化过程



如图为fraction_positive_triplets，和loss的优化过程，其中fraction_positive_triplets为所选取的一个batch的三元组中，不符合loss=0的三元组的比例。从图中可以看出，fraction_positive_triplets和loss都随着优化的次数不断减小，但loss却无法收敛，而fraction_positive_triplets可以收敛到0

7. 输出embeddings的几何模型

如右图所示为10个人的embeddings，每个embedding为128维，通过PCA降维使每个embedding变为3维，然后在三维空间中进行展示。在途中每一个点组成的簇，都表示同一个人的embedding



谢谢！