VILNIUS UNIVERSITY
FACULTY OF MATHEMATICS AND INFORMATICS
DEPARTMENT OF SOFTWARE ENGINEERING

Analysis of Loss Function for Binary Classification Problem in Deep Learning

Tikslo funkcijos analizė binarinio klasifikavimo uždaviniui spręsti naudojant gilųjį mokymąsi

Master thesis

Author: Nikolaj Kondrat
(signature)

Supervisor: prof. dr. Olga Kurasova
(signature)

Reviewer: prof. dr. Romas Baronas
(signature)

Vilnius – 2019

Santrauka

Darbe yra nagrinėjama giliojo mokymosi tikslo funkcijų taikomumas binarinio klasifikavimo problemoms spręsti. Tikslo funkcijos optimizuojančios bendrą klaidų kiekį nėra tiesiogiai adaptuotos binarinės klasifikacijos specifinių metrikų optimizavimui. Įvairūs binarinės klasifikavimo uždaviniai, turintys skirtingus vertinimo metrikos reikalavimus, nėra sprendžiami optimaliai. Siekiant išspręsti šią problemą, buvo atlikti tyrimai kurie nustato ryšį tarp tikslo funkcijos formos ir jos rezultatų priklausomai nuo skirtingų vertinimo metrikos apribojimų. Trys plačiai naudojamos tikslo funkcijos yra analizuojamos sprendžiant biometrinės veidų verifikacijos uždavinius. Remiantis tyrimų rezultatais yra surinktas praktinių patarimų rinkinys kaip pasirinkti tinkamą giliojo mokymosi tikslo funkciją pagal skirtingus veidų verifikacijos uždavinių reikalavimus.

Raktiniai žodžiai: gilusis mokymasis, tikslo funkcija, binarinis klasifikavimas, veidų verifikacija, ROC kreivė, DET kreivė

Summary

The thesis analyzes the applicability of different deep learning loss functions for binary classification problems. Misclassification error-rate based generic loss functions are not directly adapted for optimization of binary classification specific metrics. Consequently, different binary classification problems in various scenarios have custom evaluation requirements that are not being addressed. To bridge this gap, an investigation of dependencies between loss function form and its performance under different evaluation constraints is performed. Biometric face verification is chosen as a problem domain. Three commonly used loss functions are analyzed using three tasks from this domain. Based on investigation results, practical advice pack how to choose appropriate deep learning loss function for differently constrained binary classification problems from biometric face verification domain is presented.

Keywords: deep learning, loss function, binary classification, face verification, receiver operator characteristic curve, ROC, detection error tradeoff curve, DET

Contents

Introduction

Nowadays deep learning has become the first choice in many machine learning tasks from natural images classification, medical image processing to speech recognition and text translation. In essence, deep learning is re-branding name for artificial neural networks. With the development of computational power and new theoretical improvements, it has become possible to train neural networks with much more hidden layers than before. Modern graphics processing units (GPU) with thousands of computational cores are particularly suitable for fast neural network training which requires a lot of matrix multiplications. Additionally, newly introduced architectural extensions, like dropout and rectified linear unit (ReLU) activations, help to combat standard neural network problems, overfitting and vanishing gradients, which become apparent while increasing network depth.

One popular class of deep learning problems is binary classification tasks. Some examples of such tasks: particular disease identification from computed tomography (CT) scans, biometric face verification, fraud detection in the banking industry. Quite often, these problems suffer from severe class imbalance problem and simple performance metrics, like accuracy, is not suitable in such cases. Having couple fraud banking transaction cases in a list with million records, it is possible to acquire accuracy of ~99.(9)% by simply always predicting non-fraud. It is obvious that such a performance measure does not give any information about real classifier performance. For such problems more sophisticated metrics should be applied: F-measure, receiver operating characteristics (ROC) curve or its counterpart in biometric tasks - detection error tradeoff (DET) curve, area under the curve (AUC) and some others. These metrics allow to measure classifier's sensitivity and specificity at the same time and give more reliable performance evaluation.

Despite using better performance metrics for binary classification problems, objective functions being optimized during neural networks training is still misclassification error-rate based. It means that objective function is used as a proxy for desired performance metrics. For example, in [CM04] authors show that average AUC is increasing as a function of classification accuracy. However, many real-life tasks require only partial metric optimization. For example, in medical imaging computer-aided diagnosis systems operate at a fixed level of possible false positive alarms per scan, which is represented by a small section of a ROC curve. Another example - biometric security passport gates at airports operate at some predefined false match rate, which also can be mapped to a point in a DET curve. In general, different problems in various scenarios require optimization of different regions in the ROC or DET curve domain. However, optimized misclassification error-rate based objective proxy functions do not pay attention to these custom requirements. On the other hand, different loss function types are represented by different optimization patterns of binary classification metrics.

The main **objective** of this master thesis is to investigate dependencies between loss function form and its ROC or DET curve's strongest optimization subspace or point. Such analysis helps to map properties of deep learning loss functions to custom requirements of binary classification tasks.

Considering that currently deep learning is mainly used for image or video classification problems, it was decided to choose the subject area for future study from the visual domain. After an initial investigation of available datasets for different binary image classification problems, biometric face verification task was picked. Given two test face images classifier should return binary answer is it the same person or not. Such choice was motivated by the fact that there is a substantial number of publicly available labeled datasets. Considering that biometric face verification is a very popular problem in a scientific society, algorithms evaluation results reported by different research groups can be used for indirect comparison. Moreover, compared with medical image binary classification problems, it does not require special domain knowledge and does not suffer from data label noise as much.

Reaching the proposed objective should help in appropriate loss function selection for the biometric face verification tasks with specific requirements. As a study outcome, a set of rules on how to choose appropriate loss function under different constraints is expected.

To achieve the proposed objective three main tasks were identified:

1. To analyze commonly used deep learning loss functions and to choose appropriate for biometric face verification problem.
2. To analyze and choose available facial images training datasets and popular algorithm evaluation benchmarks.
3. To conduct a set of face verification experiments and using obtained results to analyze dependencies between loss function form and its performance under different binary classification evaluation constraints.

The first task is the analysis of classical and state-of-the-art deep learning loss functions and their performance on binary classification problems. Primarily this task's results are presented in the literature review part.

The second task includes collection of relevant training and testing datasets for subsequent experiments. Face verification test benchmarks with evaluation protocols requiring optimization of different binary classification metric’s regions are chosen.

The third task is to investigate possibilities to relate loss function with its ability to optimize certain subspaces of a ROC or DET curves under different specific conditions. Loss functions chosen in the first task are used. To accomplish this task extensive set of biometric face verification experiments are conducted.

Based on the main objective and described tasks, the master thesis consists of three major parts:

1. Literature review.
2. Description of the data and tools used in experiments for analysis of loss functions applied in deep learning.
3. Investigation of dependencies between loss function form and its DET curve’s optimization patterns for biometric face verification tasks by a comparative analysis of designed and conducted practical experiments.

1. Literature review

In this section research papers related to deep learning and loss functions will be review and analyzed. Literature review consists of three major parts, each describing:

1. The current state of deep learning field. This part also includes a review of problems relevant to the topic that are being solved by applying deep neural networks.
2. Loss functions applied in deep learning.
3. Optimizations of metrics designed for binary classification evaluation.

1.1. Deep learning

Deep learning is a class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification [DY14]. The main idea behind staking multiple levels of non-linear transformations is to build a hierarchical internal representation of input data, which can be useful for performing different machine learning tasks. It loosely mimics one of the theories of how the human brain might process sensory data. First attempts to construct deep neural networks (DNN) were performed in the late eighties and nineties of the previous century. However, a lack of computational power made it infeasible at that time. In addition, there were some open problems in standard error gradient backpropagation neural network training algorithm. The first problem was vanishing error gradients of the neurons with sigmoid or hyperbolic tangent activation functions. Error gradients were zeroed out effectively leaving lower layers of DNNs without training signal. The second problem was overfitting. Without proper training regularization techniques, neural networks simply learned training data samples without any generalization. Consequently, the performance on unseen data was quite poor. Some attempts to overcome this problem were performed by utilizing unsupervised pre-training techniques. Trained Deep Belief Networks (DBN) and Deep Boltzmann Machines (DBM) models were used to wisely initialize weights of DNNs and, as a result, to combat overfitting issue. However, their training procedure was quite complicated and not practically applicable to many real-life problems, for example to natural image recognition tasks.

The real breakthrough in the DNN field appeared in 2012. That year team from the University of Toronto won the ImageNet Large Scale Visual Recognition Competition (ILSVRC) using a convolutional neural network trained in a fully supervised way [KSH12]. ILSVRC is an annual competition, which evaluates algorithms for natural image recognition task on the dataset annotated for 1000 object categories. Three main components helped them to overcome all earlier mentioned problems. First, recently introduced ReLU activation function [GBB11] was used to combat the vanishing gradient problem. This function, defined as the positive part of its argument $f(x) = max(0, x)$, enables free gradient flow for neuron activations greater than zero. Second, to avoid overfitting problem they proposed a new technique, called dropout [HSK+12]. Dropout is a type of strong regularizer that prevents complex hidden units co-adaptation on the training data. This is achieved by randomly omitting each hidden unit from the network with a probability of one-half for each forward propagated training sample. It can be seen as performing kind of model averaging with neural networks. Last part of their success is the usage of Nvidia graphics processing units (GPU) with support for generic user computations. GPU architecture with a large number of low precision computing kernels is particularly suitable for neural network training, which requires a lot of vector multiplications done in parallel. By porting DNN training code to GPU they successfully overcome CPU computational power limits and establish a new trend for using graphics cards in machine learning tasks.

Roughly, all modern deep learning algorithms can be separated into three classes: supervised learning, semi-supervised learning, and unsupervised learning. The primary focus of this study is supervised learning which includes classification and regression problems. It will be presented in details in the next section.

1.1.1. Supervised learning

Supervised learning is the machine learning task which uses labeled training data for building an intelligent system solving a particular problem of interest. In this part, supervised learning in combination with deep convolutional neural networks (CNN) will be described as this type of networks most widely used in deep learning. Convolutional networks are biologically inspired trainable architecture that can learn invariant features [LKF10]. Standard layers sequence in CNN convolves two-dimensional input with a set of learnable filters, applies nonlinear transformation function to convolution output and subsamples results with pooling operation. In images raw pixels can be considered as the lowest level features, edges consisting of pixels are slightly higher-level features, edges can be assembled into parts, which also can be viewed as features, parts can be assembled into objects, and objects into scenes [LKF10]. The idea behind CNNs is that with multiple transformation layers they can learn a multi-level hierarchy of features similar to just described. The authors of [ZF14] confirmed that such hierarchical decomposition indeed happens inside CNNs. They showed that the first layer's filters resemble Gabor-like edge filters and color specific filters. It is easy to visualize the first convolutional layer as it operates on raw image pixels, however, due to the dimensionality of intermediate layers, there is no straightforward way to perform their analysis. To overcome this constraint researchers developed a method to map arbitrary layer's activations back to the image pixel space and to identify image areas that excite each neuron. By applying this method they showed from layer to layer progressively complicating features hierarchy.

Each layer of CNN gets 3D input and produces 3D output that are called feature maps. Considering color images as initial input for CNN, each color channel is a 2D feature map. Output feature map of the layer represents a particular feature extracted at all locations of the input. Feature extraction process consists of three steps.

The first step consists of filtering input feature maps with a trainable filter bank. The input is a 3D tensor, denoted by In with n 2D feature maps and each single feature map is denoted by In_i. Filter bank consists of m (defined by CNN architecture) number of filters, denoted by Ker. Each single filter, denoted by Ker_j, is 3D tensor with n (as in input tensor) 2D filter planes, denoted by Ker_{ji}. The output is also a 3D tensor, denoted by Out with m (as in kernels) feature maps and each single feature map denoted by Out_j. The filtering is done with $Out_j = b_j + \sum_i Ker_{ji} * In_i$, where $*$ is 2D discrete convolution operator and b_j is a trainable bias parameter [LKF10].

After the feature detection process is over, the second step is the pointwise application of non-linear transformation function to the previous step's outputs. Some of the possible activation functions were described in the previous section: sigmoid, hyperbolic tangent or rectified linear function.

The last step is subsampling. Feature maps obtained from the previous step are processed with a 2D sliding window by applying a *mean* or *max* function to every window. This provides reduced-dimensional feature maps that are invariant to small translations. Sliding window size and stepping stride are fixed network parameters and can be adjusted by the user. Simple input forward propagation through CNN pipeline can be seen in Figure 1.

Training of the CNN is performed using stochastic gradient descent (SGD) trying to minimize the error between the desired output and the actual network output. Output error can be calculated using various loss functions, for example, cross-entropy in classification case or square Euclidean distance in case of regression problem. For classification problems, which are the main interest of this study, mostly global accuracy optimizing loss functions are used. Gradients of the errors are computed with back-propagation technique and used to update the network weights by learning algorithm [LKF10].

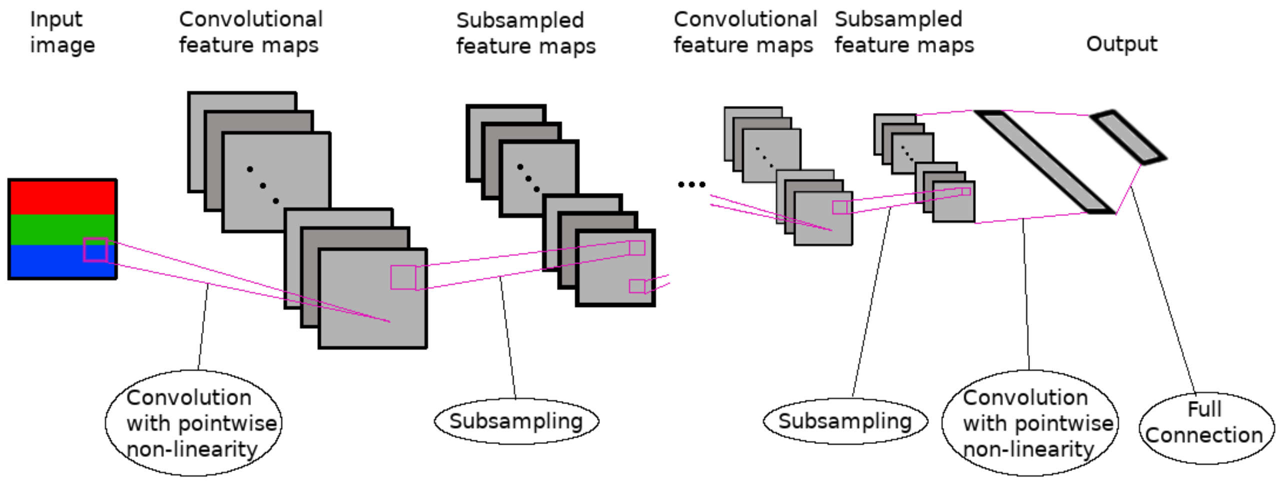
Input image
Convolutional feature maps
Subsampled feature maps
Convolutional feature maps
Subsampled feature maps
Output
Convolution with pointwise non-linearity
Subsampling
Subsampling
Convolution with pointwise non-linearity
Full Connection

Figure 1. Typical CNN architecture.

1.1.2. Tasks solved with deep learning

In this section examples of binary classification problems that are being solved by applying deep learning will be presented.

1.1.2.1. Face verification

The face is one of the main biometric modalities used for person verification together with fingerprint and iris. One of the main area for applying face verification is security systems. For example, it can be used for automatic airport security gates or police closed-circuit television (CCTV) analysis systems. Moreover, with the development of mobile technologies, it has become popular in the ordinary consumers market as well – modern cellular phones use biometric verification systems for providing access to devices. Face verification can be treated as a binary classification problem, given two test face images classifier should return answer is it the same person or not.

There are some established benchmarks for face verification algorithms evaluation. One of them is an independent face verification benchmark organized by the U.S. National Institute of Standards and Technology (NIST). It is called Face Recognition Vendor Test (FRVT) 1:1 [NIST19]. Test data is not provided to participants and evaluation is performed by NIST organization itself. It includes five different test data subsets and results are evaluated using the DET curve. The testing protocol defines fixed verification false acceptance rate for each subset and false rejection rate values at these DET curve’s points are compared between participants.

Currently, all state-of-the-art results in face recognition benchmarks are achieved with DNNs [WD18]. Finally, it is important to emphasize that evaluation protocols for face verification tasks are based on ROC or DET curve metrics. It means that the training process optimizing ordinary accuracy based metric not necessarily provides the best results. Moreover, optimization of the area under the ROC or DET curve is also not the best option, as only a single point of the curve is used for evaluation.

1.1.2.2. Lung cancer diagnosing

Another type of problems where deep learning is successfully applied is medical imaging. Computer-aided diagnosis (CAD) systems able to automatically analyze medical images of different modalities is one of the main directions in applied DNN research. For example, it can be used for detecting pneumonia from 2D chest radiographs or diagnosing Alzheimer disease from 3D magnetic resonance imaging (MRI) scans. However, there is a serious challenge that should be addressed while applying deep learning to medical image classification problems. It is required to have a lot of labeled data for successful DNN training. For example, databases used in previously described face verification tasks contain millions of images. Unfortunately, the majority of severe diseases do not have extensive training datasets with labels available. Compared to labeling natural images, producing annotations for medical images is very time consuming, labor intensive and expensive task that requires highly trained radiologists. However, there is one big exception in current medical imaging field – it is cancer research. Considering that different forms of cancer are among leading causes of death, there are many active studies and it is current mainstream in the field of computer vision applied to medical images. Cancer research organizations share datasets publicly for research purposes.

Currently, many medical imaging problems are available to the computer vision community through online competitions. One of such competitions was Lung nodule analysis (LUNA16) [STB+17] – an online challenge for lung cancer detection from computed tomography scans. Data consisted of 888 computed tomography scans taken from publicly available LIDC/IDRI database, which was annotated by four independent radiologists. Organizers provided a list of candidates with their positions. It consisted of 1557 malignant nodules and 753418 non-malignant candidates (benign nodules, segmented airways looking similar to nodules and other artifacts). It is a classical binary classification problem – to train a classifier that will be able to separate benign and malignant nodules. Evaluation protocol used free receiver operating characteristic (FROC) metric. Average sensitivity at seven predefined false positive rates from the FROC curve was used to calculate the final submission score.

Many current medical imaging problems can be cast to binary classification tasks for diagnosing particular illness from some kind of visual data. Considering typical datasets imbalance in the medical domain and unequal cost of different types of error, evaluation protocols for such tasks most often are based on the ROC curve analysis. Therefore, as in face verification case, it is not obvious how to optimize the metric of interest using standard accuracy based DNNs cost functions.

1.2. Loss functions

In this part, popular deep learning classification loss functions are presented. Unfortunately, there is no separate class of loss functions dedicated to binary classification problems that can be utilized to optimize specific to these tasks performance metrics. Therefore, generic multiclass loss functions are used.

As was stated in subsection 1.1, deep neural network training is performed by iteratively optimizing loss function. Loss function calculates average error value for each mini-batch of input samples and propagates it back to all layers. Considering that the majority of neural network training algorithms are gradient descent based, the main requirement for the loss function is its differentiability. However, sometimes non-differentiable function can be transformed and approximated with other continuous functions. Examples of such tricks will be touched on in the third part of the literature review.

1.2.1. Cross-entropy loss

The cross-entropy loss [GBC16] is the default loss for almost all classification problems where deep learning is applied. In general, cross-entropy is a broad family of loss functions that calculates negative log-likelihood of input samples. Cross-entropy can be viewed as a measure of the distance between the sample's ground truth distribution and the probability distribution predicted by a model. Intuitively, the closer two distributions are, the better the classifier's accuracy. So, during training network is optimized to match these distributions.

The cross-entropy loss for a single sample equation:

$$\mathcal{L}(x, y) = -\sum_{i=0}^{C} y_i \log p(\hat{y}_i | x)$$

where C is the class number, y is the ground truth distribution for the sample x, $\hat{y}$ is the model prediction over C classes for the sample x. Considering that the majority of classification problems are defined as single-label prediction tasks, ground truth distribution y is represented with a one-hot vector having 1 for ground truth class and 0 elsewhere. For the binary classification case the summation term can be simplified:

$$\mathcal{L}(x, y_p) = -y_p \log p(\hat{y}_p|x) - (1 - y_p)\log(1 - p(\hat{y}_p|x))$$

Where x is the input sample, y_p is the ground truth probability of a positive class and $\hat{y}_p$ is the model prediction for a positive class.

Proven simplicity and stability of cross-entropy loss together with usually acceptable results have made it the default choice for a wide variety of classification problems being solved with deep learning. However, it is still accuracy based loss function, which might be outperformed by some carefully designed functions directly optimizing binary classification related metrics like ROC or DET curves.

1.2.2. Triplet loss

The triplet loss [SKP15] is used to explicitly learn discriminative feature space embeddings for classes, thus it is not accuracy based. It does not contain the softmax layer and transforms a multiclass classification problem into a genuine pair versus impostor pair separation task.

Training input consists of a sampled batch of triples: anchor class sample, positive sample of the same class as anchor sample, negative sample of a different class with anchor sample. Training batch forward propagated through a neural network and the last layer's activations are used as a representation in a feature space. Loss function tries to minimize the distance between obtained features of anchor-positive samples pairs and, at the same time, to increase the distance between features of anchor-negative samples pairs.

The triplet loss for a single sample equation:

$$\mathcal{L} = [\|f(x^a) - f(x^p)\|_2^2 - \|f(x^a) - f(x^n)\|_2^2 + \alpha]_+$$

where x^a is the anchor sample, x^p is the positive sample, x^n is the negative sample, $f(x)$ is the feature vector of the sample x obtained by forward propagating sample through a neural network, $\|f(x)\|_2$ denotes the unit L_2 norm vector, α is the desired margin between genuine and impostor pair, $[x]_+$ operator stands for $\max(0, x)$.

The main problem of the triplet loss is that quite often it cannot be applied directly for training network from scratch. Due to proper triplet mining difficulties, it can fail to converge to a local minimum with ordinary learning rates. Therefore, a typical workflow might include softmax classification pre-training and subsequent triplet loss fine-tuning.

The authors of [SKP15] successfully applied triplet loss to the face verification problem and obtained several benchmark results that are still close to the current state-of-the-art. However, it is not limited to biometric problems and, for example, was used for content-based image retrieval task by researchers from Yahoo [DNH+18].

1.2.3. Magnet loss

The magnet loss [RPD+16] is designed by Facebook researchers and is used to enhance intra-class and inter-class similarity measure. Generic cross-entropy loss applied to softmax outputs takes into account information provided by labels only. Therefore, it assumes unimodal distribution for each class and does not care about internal variations. Magnet loss partition single class feature representation space into an arbitrary number of clusters and assign each sample to the nearest cluster center. It does not require any additional information as clustering is unsupervised technique. For example, dogs and cats are quite often photographed together with their owners. Magnet loss might detect this trend and optimize clusters of cat and dog labeled samples in an appropriate way: instances of a cat with an owner and a dog with an owner will be placed closer to each other in the representation space than instances of single cat and dog. It can be interpreted as an unsupervised introduction of virtual generalizing class "pet with human". Such an approach helps to discover shared structures between different classes.

During training, a representation feature vector is extracted from a sample and assigned to a cluster with the nearest center. The magnet loss function tries to minimize the distance to this cluster and at the same time to increase the distance to the selected number of nearest clusters of impostor classes.

The magnet loss for a single sample equation:

$$\mathcal{L} = \left\{ -log \frac{e^{-\frac{1}{2\sigma^2}\|f(x)-\mu(f(x))\|_2^2-\alpha}}{\sum_{c \neq C(x)} \sum_{k=1}^{K} e^{-\frac{1}{2\sigma^2}\|f(x)-\mu_k^c\|_2^2}} \right\}_+$$

where $f(x)$ is the feature representation of the input sample, $\|f(x)\|_2$ denotes the unit L$_2$ norm vector, σ^2 is the variance of all examples away from their respective cluster centers, μ_k^c is the cluster center, $\mu(f(x))$ is the function that assigns $f(x)$ to the closest cluster center, c is the set of all class labels, $C(x)$ is the ground truth label for the input sample, K is the number of clusters assigned to each class, α is the desired cluster separation gap, $\{\cdot\}_+$ is the hinge function.

One drawback of this method is its requirement to recalculate periodically cluster centers as training process evolves. However, the authors state that it should not be done frequently. Moreover, clusters count is a hyperparameter that should be set before training start and it brings additional complexity, as it might be not obvious how to choose an appropriate value.

The authors of [RPD+16] successfully applied magnet loss for fine-grained natural image recognition, object attributes classification and hierarchy recovery tasks. All tasks imply the model's ability to capture additional intra-class representation variations that are not available from a single label information. Despite that it is primarily designed for multiclass problems, it can be adapted to binary classification cases too. For example, lung cancer dataset from [STB+17] in addition to binary labels has malignancy level evaluations for cancer-positive samples. Such information can be used for establishing additional clusters within each class and potentially improve classifier performance on difficult samples near the decision boundary.

1.2.4. Additive angular margin loss

The last loss function presented in this part is the additive angular margin loss [DGZ18]. It is recently introduced loss that achieves state-of-the-art results in face verification and recognition tasks. Despite the fact that originally it was designed for dealing with biometric problems, it can be applied to any other classification task. The main idea of this loss is to embed all class representations on a hypersphere of the chosen radius. The loss function form ensures that all single class samples should be compactly placed around mean class representation. This is achieved by directly minimizing angle between mean class representation feature vector and feature vectors of samples belonging to that class. At the same time, inter-class distance maximization on a hypersphere is guaranteed by applying standard softmax function. In general case, to obtain mean class representation it is required to collect statistics over class samples what can be computationally expensive. However, [DGZ18] authors solved this problem by using L_2 normalized class column vectors of the classification layer as mean class representations. Therefore, angles between weight column vectors and sample penultimate layer's activation vector are optimized during training. In general, this loss can be viewed as an extension of standard cross-entropy over softmax.

The additive angular margin loss for a single input sample equation:

$$\mathcal{L} = -log \frac{e^{s(\cos(\theta_{y_t}+\alpha))}}{e^{s(\cos(\theta_{y_t}+\alpha))} + \sum_{j=1, j\neq y_t}^{C} e^{s*\cos(\theta_j)}}$$

where θ_j is the angle between the j^th column of the classification layer's weight matrix W and the output vector of the penultimate network's layer for the input sample, α is the angular margin hyperparameter, C is the number of classes, s is the hypersphere radius hyperparameter and y_t is the ground truth class label of the input sample. Margin hyperparameter α adds an angular penalty to ensure intra-class compactness of representations.

The authors of [DGZ18] successfully applied this loss to a face verification problem. For training, they used labeled dataset consisting of about 3.5 million images with more than 80000 identities. At first, they converted face verification problem to multiclass classification problem by converting each dataset's identity to a separate class. Then, standard training procedure of ResNet-like architecture network with additive angular margin loss was held. After training, the last classification layer was cut and the penultimate layer outputs were used to calculate the distance between test samples to decide whether it is the same person or not.

1.3. Binary classification metric optimization

Despite the fact that deep learning has no separate algorithms for optimization of binary classification specific performance metrics, there are some attempts to construct such procedure in adjacent machine learning fields. One of the main challenges for directly using ROC, DET, precision-recall curves metrics in optimization procedure is the fact that they are non-decomposable objectives. They cannot be easily expressed in terms of a single training point. This undesirable property disables mini-batch optimization methods. Moreover, most often they are non-smooth functions, making gradient descent based method not applicable. However, quite often it is possible to find approximation with desirable properties. In addition, some tricks changing training sampling strategy were proposed to overcome the non-decomposability issue. In this part, studies for direct optimization of such metrics will be presented.

1.3.1. Linear models

One of the first attempts to optimize the ROC curve metric directly was presented in [MDC+02]. In this study, the authors tried to predict whether telecom client is willing to switch service provider or not. Their task was defined as achieving the required false rejection rate (FRR) at some fixed predefined false acceptance rate (FAR). It can be seen as a similar problem to described earlier Face Recognition Vendor Test (FRVT) 1:1 case.

To solve this task they proposed iterative re-training strategy. At first, they trained logistic regression classifier as usual. After that, they ranked all training samples by scores obtained from the trained model and calculated thresholds for achieving task defined FRR and FAR rates using this ranking. Having two different thresholds, their main objective was to make them equal. To achieve this, samples scored in between two thresholds were selected and classifier was retrained from scratch by up-weighting selected samples.

Such an approach has two main drawbacks. First, they achieved required FRR and FAR rates on a training set and did not provide any information about testing on a held-out dataset. Second, they used a simple linear model with small training time, however, in case of DNNs, it can be prohibitively expensive to use iterative strategy.

In [CM04] authors evaluated the hypothesis that training to minimize classification error rate (or maximize accuracy) is the same as ROC AUC optimization. They discovered that for the same model’s classification accuracy rate there could be quite different AUC values depending on the test set samples ranking. However, these two measures are interdependent: in case of even data distributions, where a number of positive and negative samples is equal, average AUC is the same as classifier accuracy. For uneven distributions average AUC is a monotonic function of the accuracy. Moreover, authors discovered dependency that more uneven distributions have bigger AUC variance. Unfortunately, for many problems where deep learning is applied, for example, medical imaging, it is more realistic to expect uneven data distributions. So, it worth to try to construct specifically AUC-targeted loss function in order to obtain better than average results.

In [KTT+12] authors propose online training dataset storing strategy to train a linear classifier model to directly optimize AUC. In general, AUC optimization requires full pairwise ranking matrix construction. The authors proposed the idea of storing all previously extracted features together with a pairwise distance matrix in memory. As the new training sample comes, the algorithm extracts features, calculates distances to all previously seen data points and updates both the pairwise ranking matrix and stored features database. Then classifier parameters are updated using a distance matrix as a parameter of quadratic function approximating AUC objective. In case of the substantially large training dataset, only some fraction of the last training samples can be stored without any performance degradation.

This is a classical machine learning approach when feature extractor and classifier are separate and independent instances. Feature extractor is fixed during the classifier training process. On the contrary, the deep learning approach is joint, end-to-end, training of feature extractor and classifier. Such settings invalidate the idea of storing a matrix of pairwise distances of training samples as this matrix is not constant anymore and should be updated after each model’s update. This is not viable as a DNN mini-batch training procedure performs hundreds of thousands of model parameters update iterations.

1.3.2. Deep learning models

Considering that deep learning has become popular quite recently, there are not so many published studies about optimizations of non-decomposable objectives with DNNs.

The authors of [SSZ+16] tried to overcome the limitations of the method proposed in [KTT+12]. They performed online optimization of average precision metric, which represents area under the precision-recall curve. The main contribution of this work was extending further the idea that calculation of full training dataset pairwise rankings is not necessary. Results of [KTT+12] showed that only some fraction of the last training samples could be used as an approximation of the full dataset. In case of deep learning, the main problem with storing a pairwise distance matrix is that it constantly changes with model parameters updates. Therefore, they decided to approximate it just with a single training batch. It is a purely online method, which does not require any additional previously used data. Performed tests showed that even such crude approximation produces slightly better results than cross-entropy loss function used as a proxy for the desired metric. Moreover, the authors performed a synthetic test by training DNN using a dataset with intentionally contaminated labels. The improvement in performance became even more obvious. Most probably, such behavior can be justified by the fact that approximation of the dataset pairwise ranking matrix from a single batch produces a noisy signal by itself and additional noise is not so significant. This observation can be useful for medical imaging problems, where labeling quality quite often is a real issue.

In [ESM+17] researchers from Google described methods for direct optimization of maximum recall at fixed precision (R@P) and average precision metrics for information retrieval systems. They proposed a method to approximate the non-differentiable R@P calculation equation by changing exact true positive (TP) rate with lower bound TP rate and exact false positive (FP) rate with upper bound FP rate. Such substitution produces a relaxed problem of R@P optimization, which is concave lower bound of the original R@P. After that, they optimized R@P by applying Lagrange multiplier theory with fixed precision used as a constraint. Considering that every term in transformed problem definition is differentiable, standard SGD optimizer was used. Average precision was optimized by dividing precision values interval into a set of discrete anchor points and performing summation of joint R@P optimizations for each anchor point. The reported average precision improvement for ILSVRC benchmark is around 1%, which is quite substantial for large-scale recognition problems. However, the proposed algorithm looks highly computationally demanding and might be not feasible outside of companies like Google, as authors reported that training required a cluster of 50 hi-end GPUs running for three days.

1.4. Literature review summary

During literature review popular deep learning loss functions and results evaluation metrics applied in binary classification problems were analyzed. In addition, studies attempting to design algorithms for direct binary classification metrics optimization were presented.

Unfortunately, some aspects left untouched in reviewed research papers and consequently some potential places for improvements were identified:

1. There is no dedicated class of deep learning loss functions for binary classification tasks. At the same time, the lack of comparative analysis of presented loss functions in terms of metrics that are more suitable for binary classification problems was identified. Performing such analysis will be the main target of this master thesis.
2. Most of direct binary classification metrics optimization methods have come from classical machine learning with separate feature extractor and classifier entities and are not suitable for end-to-end trainable models like DNNs. Investigation of how to adapt these algorithms to deep learning will be left for future work.

2. Data and tools

In this section face verification datasets and deep learning software framework used for experiments will be described.

2.1. Labeled Faces in the Wild (LFW) dataset

LFW [HRB+07] is probably the oldest and the most widely used dataset for face verification benchmarking. It was publicly released by researchers from the University of Massachusetts in 2007 together with algorithms evaluation protocol.

The dataset contains 13233 photos of 5749 different individuals, mostly celebrities, sportsmen and politicians. It was collected from different internet sources and includes pictures with a wide variability of poses, facial expressions, and illumination conditions. As can be seen, it is quite a small dataset with most individuals having only a single image. Some images contain multiple faces, however, only a single person was annotated for each image.

Standard face verification protocol includes 3000 genuine pairs (positive label) and 3000 impostor pairs (negative label) for comparison. Performance is evaluated and compared between different algorithms by obtaining the value of a DET curve’s equal error rate (EER) point. Unfortunately, there is a small number of discovered errors in benchmark pairs, but for compatibility with previously reported results authors insist that it should be used “as is”.

Despite that this is one of the oldest benchmarks and now can be considered as a solved problem (EER of more than 99.5% is achieved by many participants [WD18]), it is still wildly used as a reference. In this study, LFW is used as a test dataset for comparison of different deep learning losses evaluating the EER DET point. Images were preprocessed by running face detection algorithm for obtaining tight crops around faces. Considering that it is quite a small dataset, preprocessing results were manually verified and corrected in case of wrong person detection or failure.

2.2. Multiple Encounter Dataset II (MEDS-II) dataset

MEDS-II [FOW+11] is another standard face verification dataset. It was publicly released by National Institute of Standards and Technology (NIST) in 2011.

The dataset contains 1216 photos from 518 different individuals. It was collected in the Federal Bureau of Investigation Data Analysis Support Laboratory and includes mugshots of arrested people, mostly frontal. There is low pose, facial expressions, and illumination conditions variability. In addition, MEDS-II dataset is strongly gender biased – the vast majority of individuals are males.

Despite that MEDS-II is even smaller than LFW dataset, its evaluation protocol requires the construction of all possible image pairs, producing a much bigger test set. However, considering that the majority of individuals have only one or two photos, genuine pairs (positive label) number is three orders of magnitude less than impostor pairs (negative label) number. Performance is evaluated and compared between different algorithms by obtaining FRR value of DET curve at the fixed 0.001% FAR point.

In this study, MEDS-II is used as a test dataset for comparison of different deep learning losses evaluating FRR value at the 0.001% FAR point of a DET curve. Images were preprocessed by running face detection algorithm for obtaining tight crops around faces. Considering that it is quite a small dataset, preprocessing results were manually verified and corrected in case of detection failure.

2.3. Microsoft One Million Celebrity (MS-Celeb-1M) dataset

MS-Celeb-1M [GZH+16] is probably the biggest publicly released dataset for training and evaluating face recognition algorithms. It was released by Microsoft Corporation in 2016.

The dataset contains about 10 million photos from 100000 different individuals. It was automatically collected from different internet sources and includes pictures with a wide variability of poses, facial expressions, and illumination conditions. Despite that after initial collection it was cleaned-up and verified using different algorithms, it still contains a substantial level of label noise. Some images contain multiple faces, however, only a single person was annotated for each image.

Dataset authors propose to use precision-coverage metric in evaluation protocol. This metric is more suitable for face identification problems, which can be viewed as ranking tasks. However, they also state that other metrics are welcomed to report. Considering that the main focus of this study is binary classification problems and their evaluation using ROC or DET curves, it was decided to use metric from Face Recognition Vendor Test (FRVT) provided by NIST [NIST19] for Wild dataset images. It evaluates FRR value at the $1 \cdot 10^{-4}$ FAR point of a DET curve using all possible image pairs. The difference of fixed FAR point value with MEDS-II dataset (0.01% versus 0.001%) can be explained by problems complexity – unconstrained photos from different sources versus strictly regulated mugshot images.

In this study, only small subsets of MS-Celeb-1M dataset is used. Such reduction is done due to computational and time constraints, as a full dataset training lasts around two weeks using workstation with two top-level Nvidia Titan V GPUs. The MS-Celeb-1M dataset is used for both neural networks training and different deep learning losses performance evaluation. It was decided to construct a test subset by choosing 5000 images of 1000 identities that overlap with LFW dataset. The constructed test set has much greater genuine pairs (positive label) and impostor pairs (negative label) number than two previous datasets and can be used as a more reliable results source. After some initial experiments, it was determined that the optimal training dataset should include 750000 images of 15000 different identities. Such training data amount gives the ability to identify clearly differences between deep learning losses performance. Still, some experiments with smaller training datasets (375000 and 250000 images) are presented for comparison. Validation subset was constructed in the same way as the test subset by choosing 5000 images of another 1000 identities that overlap with LFW dataset. Images were preprocessed by running face detection algorithm for obtaining tight crops around faces. Considering dataset size, manual verification of preprocessing results wasn’t performed.

2.4. PyTorch deep learning framework

In this section brief overview of PyTorch deep learning framework [PGC+17], that was used for experiments in this study, will be given.

PyTorch is deep learning package for fast and flexible experimentation developed at Facebook. It is written in C++/CUDA and Python programming languages having deep integration with NumPy scientific computing package. Main features that give the ability to conduct fast experiments using this framework is automatic differentiation capabilities and dynamic execution graph construction. The first one frees machine learning engineers and data scientists from the tedious task of analytical derivatives calculation while implementing non-standard framework extensions. The second one makes the backpropagation process, which is in the heart of neural network training, transparent by providing an interface to easily access data for inspection at any execution step. It drastically simplifies new algorithm debugging process.

Due to its modularity, it is straightforward to extend it if some additional functionality is needed by the task. Main framework unit is data tensor. There are many tensor operations provided out of the box, which cover almost all standard machine learning tasks. In this study neural network architectures that mainly used these types of tensor operations were used:

- 2D discrete convolution operation. For input spatial dimensionality reduction convolutions with a stride of 2 were used, so no additional pooling operations were involved.

- Linear data transformation, which performs simple two matrices multiplication for changing linearly input data representation.
- Parametric rectified linear unit (PReLU) [HZR+15] nonlinearity operation, for introducing nonlinearity between linear data transformations.
- Batch normalization [IS15], for reducing the impact of data distribution changes in randomly sampled mini-batches.

PyTorch framework has built-in support for running training process on the modern Nvidia GPUs. Due to GPU architecture peculiarity, it gives around 25 times faster training and inference speed than top-level CPUs.

Considering that convolutional neural network (CNN) classifier training performs best when original input data is mixed with artificially slightly distorted samples, online data augmentation transformers extension was implemented for experiments. Data augmentations include random rotations, flipping, shifting, scaling, cropping, blurring, brightness and contrast stretching of samples. Geometric transformations help to train models robust to face pose variations, while color transformations address the variability of illumination between samples. Taking into account that two test datasets, LFW and MS-Celeb-1M, includes images taken in unconstrained environments, it should noticeably improve results.

For evaluating DET curve's points of interest training loop extension was implemented. After a chosen number of forward-backward propagation iterations training is automatically paused and DET curves from described earlier datasets are calculated. With a configurable interface, it is possible to get programmatically specified curve's values and evaluate test set performance. Such an extension can be used not only for evaluation purposes but also for training process controlling. Adjusting of learning rate or dynamical samples reweighting can be performed according to validation or training DET curves results. Considering that the DET curve is just inverted ROC curve's representation, it is possible to use this extension for binary classification problems evaluated with ROC metric.

3. Loss functions analysis

In this section experiments conducted during the investigation of dependencies between deep learning loss function form and its ROC or DET curve behavior will be described and results analyzed. Three loss functions, presented earlier in the literature review part of this study, will be used: cross-entropy loss [GBC16], triplet loss [SKP15], and additive angular margin loss [DGX+18]. Such choice is made after evaluating time and resources constraints and motivated by the observation that these three loss functions are the most popular in generic binary classification and biometric face verification problems. Standard implementation of cross-entropy loss function from PyTorch library was used. Triplet and additive angular margin loss functions were implemented following specification from original papers.

In the following subsections, figures with results from conducted experiments are presented. It is the DET curve metric that is used for a binary classifier evaluation, in particular, in biometric verification tasks. The x-axis represents false acceptance rate (FAR), in other words, how many negative samples were classified as positives. The y-axis represents false rejection rate (FRR), in other words, how many positive samples were classified as negatives. Three points taken from described earlier benchmark evaluation protocols are used for a comparison of curves: the EER point (LFW benchmark), FAR=0.01% (MS1M benchmark) and FAR=0.001% (MEDS benchmark). The lower the curve is (lower FRR), the better the results are. If curves' behavior pattern is not changing between FAR=0.001% and FAR=0.01%, results for these points are aggregated under the name of "low FAR region".

3.1. Convolutional neural network architectures

In this section overview of CNN architectures used in this study will be presented. It was decided to use ResNet [HZR+16] CNN architecture with different networks depth. ResNet CNN architecture provides current state-of-the-art results in many computer vision benchmarks and is very popular among deep learning researchers and engineers. It has an appealing property of fast convergence and gives the ability to use higher learning rates drastically reducing training time. Considering time and GPU resources constraints, it is probably the most optimal choice for experiments.

The main advantage of this type of CNN architecture is its ability to train very deep networks avoiding vanishing gradients problem. Vanishing gradients is an old-known problem occurring during the training process – gradients back propagated from the top decrease exponentially with the number of layers during backward pass. Consequently, bottom layers do not get a training signal and the learning process does not converge. The solution that He and his colleagues proposed was the introduction of layer blocks and skip connections. Skip connection gives the ability to skip a block of layers and to transfer forward propagated signal and back propagated gradients through a simple identity connection additionally to an original signal. With such trick, gradients are not decreasing so heavily and it is possible to train even 1000 layers CNNs. However, in [VWB16] researchers empirically show that effective network depth still stays around 20-35 layers, all other layers are skipped.

For the main experiments, it was decided to use 21-layered ResNet architecture (ResNet21). Such network's depth stays within the effective range reported in [VWB16] and requires an affordable amount of time and GPU resources for training. Also, it was decided to conduct an additional set of experiments with a smaller network for identifying possible training data overfitting cases. For these experiments 13-layered ResNet architecture (ResNet13) was used. High-level graphical representation of architectures can be seen in Figure 2. Detailed blocks structures are presented in Appendix 1.

The ResNet21 network has 25 convolutional layers with an effective depth of 21 layers. This difference occurs due to pointwise convolutions with one by one kernels in skip connections of residual blocks with spatial dimension reduction. In addition, it has two fully connected layers in the network's output block. This is slightly different from the original ResNet architecture where the first fully connected layer is replaced with global average pooling. Empirical results show more accurate predictions with this modification. Neuron activations from convolutional and fully connected layers are passed through batch-normalization layers. PReLU nonlinearity is used for non-linear data transformation between layers. For 2D input dimensionality reduction, convolutional layers with stride two are used, effectively reducing input size by a factor of two. As an additional regularizer dropout layer [HSK+12] is used in the network's output block, preventing complex hidden units co-adaptation on the training data.

The ResNet13 network is constructed from ResNet21 network by excluding four residual blocks. It has 17 convolutional layers with an effective depth of 13 layers.

In all experiments, CNNs were trained by applying SGD optimizer with Nesterov momentum. Additional weight decay regularization of $5 \cdot 10^{-5}$ is applied to convolutional and fully connected layers' weights.

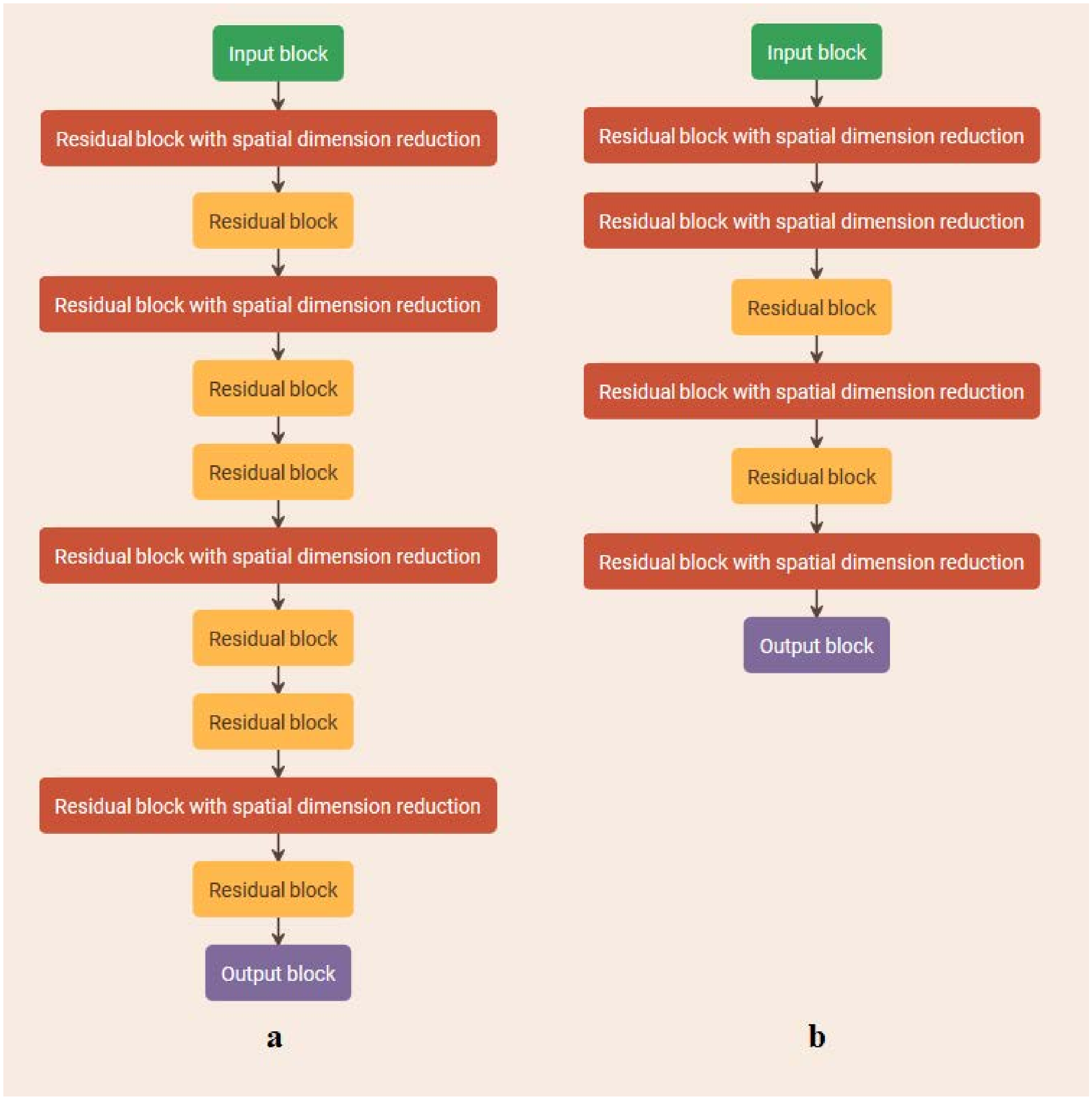
Input block
Residual block with spatial dimension reduction
Residual block
Residual block with spatial dimension reduction
Residual block
Residual block
Residual block with spatial dimension reduction
Residual block
Residual block
Residual block with spatial dimension reduction
Residual block
Output block
a
Input block
Residual block with spatial dimension reduction
Residual block with spatial dimension reduction
Residual block
Residual block with spatial dimension reduction
Residual block
Residual block with spatial dimension reduction
Output block
b

Figure 2. ResNet architectures: a) ResNet21, b) ResNet13

3.2. ResNet21 architecture training with MS-Celeb-1M large dataset

The first set of experiments was done using baseline ResNet21 CNN architecture and MS-Celeb-1M large dataset, consisting of 750000 images of 15000 individuals. Training process duration was set to 500 epochs (one epoch equal to complete pass through a given training dataset). Considering time constraints, it was decided to apply early stopping [Pre12] technique based on results from validation dataset. Early stopping helps to combat overfitting and saves time in case of training process stagnation – if validation results are not improving for a predefined number of iterations, then the training loop is stopped. Three described earlier datasets were used for results evaluation and comparisons using the DET curve.

For the first experiment, the cross-entropy loss function was chosen as it is the default loss for almost all classification problems where deep learning is applied. The ResNet21 model was trained with mini-batch size of 256 samples using a base learning rate of 0.1. Using the information provided by an early stopping criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). Further learning rate reduction did not bring any improvements. The complete experiment required 120 training epochs and took around 30 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 3 in blue color (full-sized view can be found in Appendix 2: Figure 16). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 14, 15.

The second experiment was performed with the triplet loss. Initial attempt confirmed proposition stated in [SKP15] that the majority of randomly sampled triplets are too trivial. That way trained network does not get useful learning signal and converges to a bad local minimum. To overcome this issue and to improve results, online hard triplet mining strategy proposed in [SKP15] was implemented. The ResNet21 model was trained with mini-batch size of 32 triplets using a base learning rate of 0.1. Unfortunately, triplet loss convergence rate is quite slow and early stopping technique was almost useless for the base learning rate. Nevertheless, it helped to reduce training time after learning rate reductions. The training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 549 training epochs and took around 147 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 3 in green color (full-sized view can be found in Appendix 2: Figure 16). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 14, 15.

The last experiment was ResNet21 architecture training with the additive angular margin loss. According to [DGX+18] margin hyperparameter was set to 0.5 value. CNN was trained with mini-batch size of 256 samples using a base learning rate of 0.1. As well as the cross-entropy loss, the additive angular margin loss also can be successfully combined with early stopping technique. Using the information provided by this criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). Further learning rate reduction did not bring any improvements. The complete experiment required 156 training epochs and took around 33 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 3 in red color (full-sized view can be found in Appendix 2: Figure 16). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 14, 15.

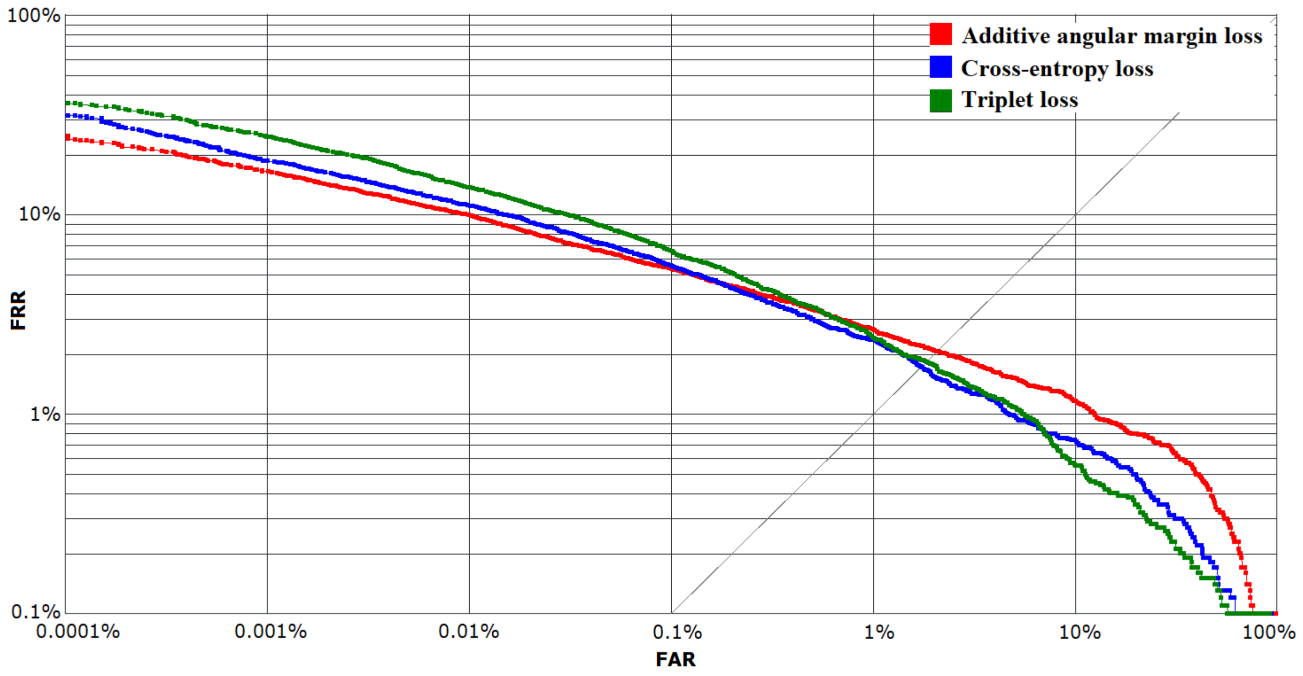

Figure 3. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

By analyzing DET curves in Figure 3, it can be noticed that different losses behave differently in particular regions. Here and in future comparisons, MS-Celeb-1M test dataset will be used as the source of the most reliable results. Such a decision is motivated by the fact that it has substantially bigger pairs number and statistically should be more reliable. By looking at the low FAR region (0.00001%–0.01%) it can be noted that the additive angular margin loss performs the best. The cross-entropy loss is in the middle and the triplet loss is the worst one. This region is particularly important for MS-Celeb-1M and MEDS-II test datasets. However, comparing curves at the EER point, situation changes – cross-entropy and triplet losses have similar results and the additive angular margin loss performs the worst. This point is important considering LFW dataset evaluation protocol.

By comparing plots between different datasets, it can be stated that results agree between themselves. Considering LFW DET curves (Appendix 2: Figure 14), it should be clarified that thin straight horizontal line in the low FAR region means that there is no information to calculate values from. Therefore, it cannot be interpreted in favor of the cross-entropy loss and should not be used at all for evaluating this particular curve's region. Final results, evaluated according to described earlier dataset protocols, are presented in Table 1.

Table 1. Evaluation protocol results: ResNet21 CNN trained with MS-Celeb-1M large dataset.

	Cross-entropy loss	**Triplet loss**	**Additive angular margin loss**
LFW (EER)	0.5%	0.6%	0.65%
MEDS-II (FRR@FAR=0.001%)	4.626%	5.397%	3.356%
MS-Celeb-1M (FRR@FAR=0.01%)	11.18%	13.76%	9.99%

3.3. ResNet21 architecture training with MS-Celeb-1M medium dataset

The second set of experiments was done using the same baseline ResNet21 CNN architecture and MS-Celeb-1M medium dataset, consisting of 375000 images of 7500 individuals. The same training process and results evaluation protocols were applied as in the first set of experiments.

For the first experiment, the cross-entropy loss function was chosen again. The ResNet21 model was trained with the same settings of SGD optimizer's hyperparameters as in the first set of experiments. Using the information provided by an early stopping criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 300 training epochs and took around 32 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 4 in blue color (full-sized view can be found in Appendix 2: Figure 19). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 17, 18.

The second experiment was performed with the triplet loss. Hyperparameters settings were copied from the corresponding experiment with MS-Celeb-1M large dataset. The training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 657 training epochs and took around 157 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 4 in green color (full-sized view can be found in Appendix 2: Figure 19). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 17, 18.

The last experiment was ResNet21 architecture training with the additive angular margin loss. Hyperparameters settings were copied from the corresponding experiment with MS-Celeb-1M large dataset. Using the information provided by an early stopping criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 132 training epochs and took around 29 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 4 in red color (full-sized view can be found in Appendix 2: Figure 19). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 17, 18.

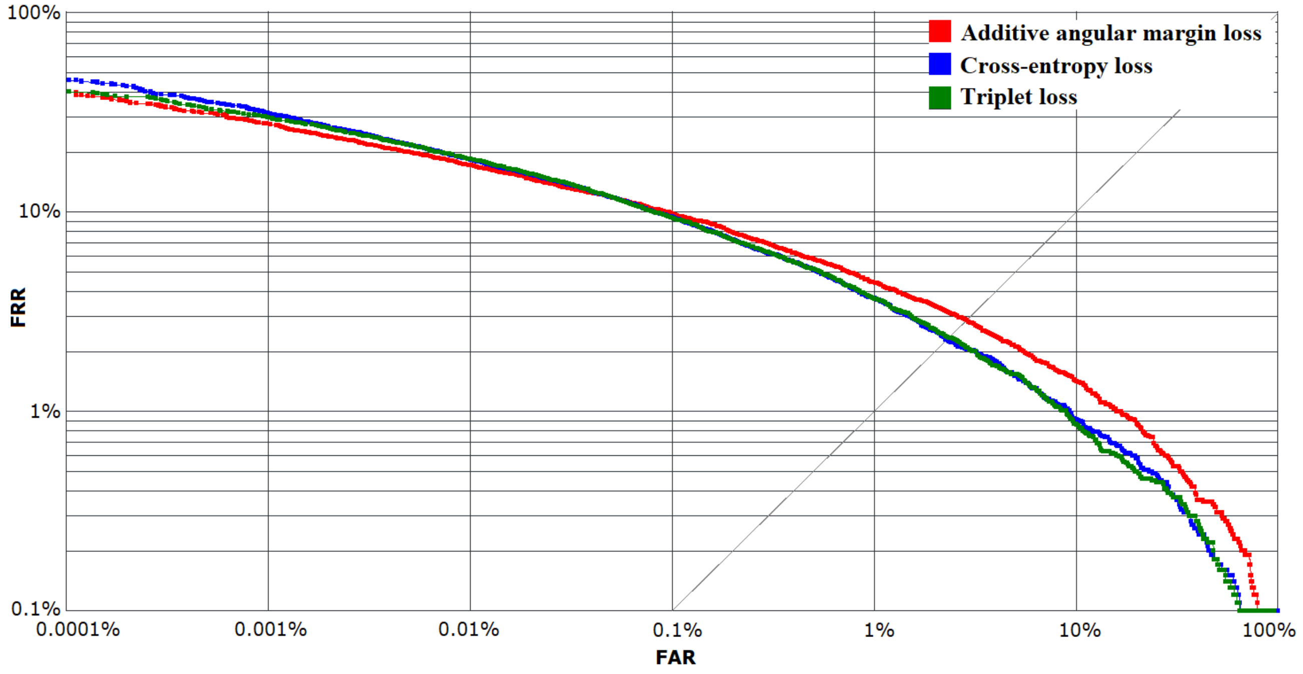

Figure 4. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M medium dataset.

By analyzing DET curves in Figure 4, it can be noticed that the additive angular margin loss is still the best performing one in the low FAR region. However, the difference is not so obvious anymore. This tendency leads to a hypothesis that the additive angular margin loss requires more training data than cross-entropy and triplet losses to converge to a good local minimum. Moreover, the difference between triplet and cross-entropy losses also has become insignificant in this region. Behavior pattern at the EER point is the same as in experiments with MS-Celeb-1M large dataset – cross-entropy and triplet losses have similar results and the additive angular margin loss performs the worst. By comparing plots between different datasets, it can be stated that results agree between themselves. Final results, evaluated according to described earlier dataset protocols, are presented in Table 2.

Table 2. Evaluation protocol results: ResNet21 CNN trained with MS-Celeb-1M medium dataset.

	Cross-entropy loss	**Triplet loss**	**Additive angular margin loss**
LFW (EER)	0.75%	0.8333%	0.9667%
MEDS-II (FRR@FAR=0.001%)	13.74%	8.254%	7.347%
MS-Celeb-1M (FRR@FAR=0.01%)	18.43%	18.56%	17.24%

3.4. ResNet21 architecture training with MS-Celeb-1M small dataset

The third set of experiments was done using the same baseline ResNet21 CNN architecture and MS-Celeb-1M small dataset, consisting of 250000 images of 5000 individuals. The same training process and results evaluation protocols were applied as in the first set of experiments.

The first experiment was conducted with ResNet21 architecture and the cross-entropy loss function. The training was performed with the same settings of SGD optimizer's hyperparameters as in the first set of experiments. Using the information provided by an early stopping criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 305 training epochs and took around 29 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 5 in blue color (full-sized view can be found in Appendix 2: Figure 22). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 20, 21.

The second experiment was performed with the triplet loss. Hyperparameters settings were copied from the corresponding experiment with MS-Celeb-1M large dataset. The training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 540 training epochs and took around 121 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 4 in green color (full-sized view can be found in Appendix 2: Figure 22). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 20, 21.

The last experiment was ResNet21 architecture training with the additive angular margin loss. Hyperparameters settings were copied from the corresponding experiment with MS-Celeb-1M large dataset. Using the information provided by an early stopping criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 63 training epochs and took only 5 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 4 in red color (full-sized view can be found in Appendix 2: Figure 22). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 20, 21.

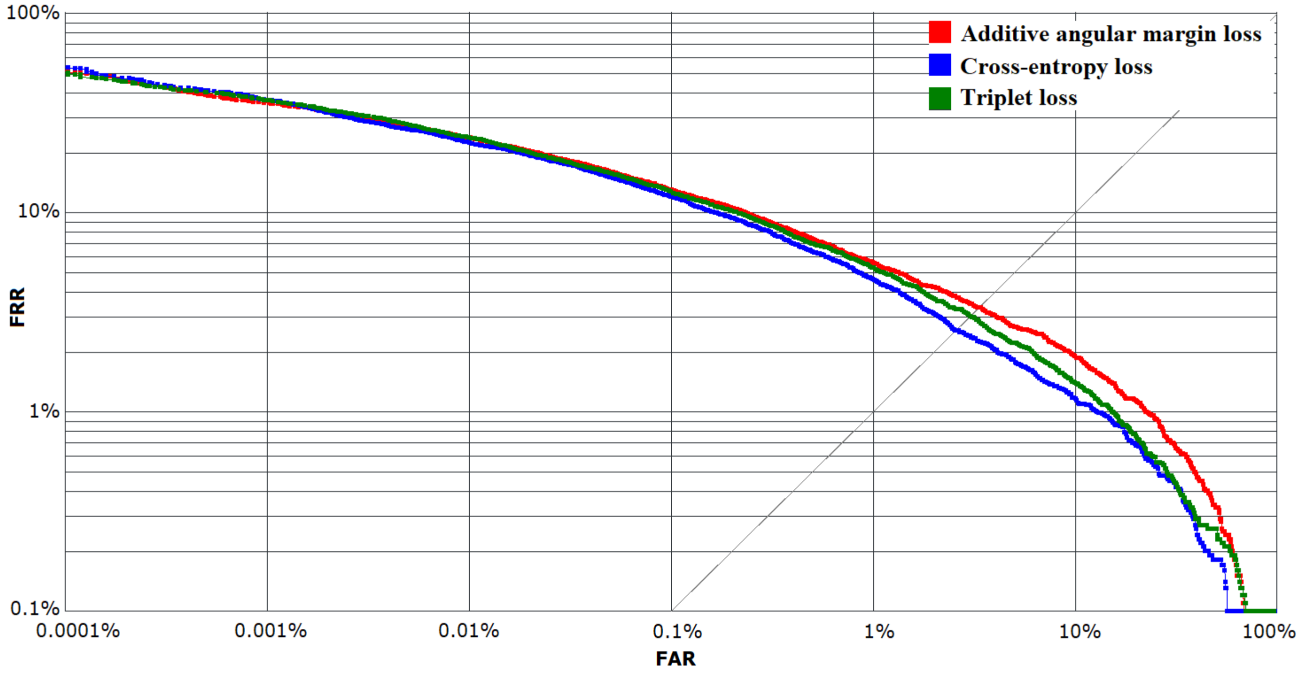

Figure 5. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M small dataset.

By analyzing DET curves, it can be noticed that the additive angular margin loss completely loses its advantage in the low FAR region – all losses perform almost identically. This observation confirms proposed earlier hypothesis that additive angular margin loss requires more training data than cross-entropy and triplet losses to converge to a good local minimum. By analyzing curves at the EER point, it can be noticed that the cross-entropy loss superiority has become more obvious in the setting of scarce training data. By comparing plots between different datasets, it can be stated that results agree between themselves. Final results, evaluated according to described earlier dataset protocols, are presented in Table 3.

Table 3. Evaluation protocol results: ResNet21 CNN trained with MS-Celeb-1M small dataset.

	Cross-entropy loss	**Triplet loss**	**Additive angular margin loss**
LFW (EER)	1.033%	1.233%	1.25%
MEDS-II (FRR@FAR=0.001%)	12.29%	12.47%	13.38%
MS-Celeb-1M (FRR@FAR=0.01%)	22.57%	23.99%	23.83%

3.5. ResNet21 architecture training: dataset size impact

In this section results from three previous experiment sets are aggregated by loss functions and presented for analysis of training dataset size impact. In Figure 6 (full-sized view can be found in Appendix 2: Figure 25) all three experiments with the cross-entropy loss are presented for MS-Celeb-1M test set. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 23, 24. The cross-entropy loss demonstrates almost linear dependency between dataset size and curve's improvement for the whole region from the zero FAR point till the ERR point. This is a highly desirable property for generic classification loss. However, for some specific problems, like tasks in this study, it might be not that relevant to optimize the whole curve and to prefer only some particular regions.

Some discrepancies can be seen for MEDS-II dataset experiments in Appendix 2: Figure 24. The difference between DET curves for small and medium-size training datasets is negligible. One of the possible reasons is that MEDS-II dataset, collected in constrained environments, is substantially different from LFW and MS-Celeb-1M datasets, collected from different online sources. Considering the random model initialization at the beginning of the training process and highly multidimensional loss function surface, it is feasible that optimizer has converged to a bad local minimum for that particular type of input data.

In Figure 7 (full-sized view can be found in Appendix 2: Figure 28) all three experiments with the triplet loss are presented for MS-Celeb-1M test set. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 26, 27. The triplet loss demonstrates monotonically increasing improvement moving from the zero FAR point to the ERR point with training set size growth. Such behavior can be beneficial in tasks like LFW face verification where evaluation is performed using the EER DET curve's point.

Some minor discrepancies again can be noticed for MEDS-II dataset in Appendix 2: Figure 27 at the EER point for models trained with medium and large MS-Celeb-1M training datasets.

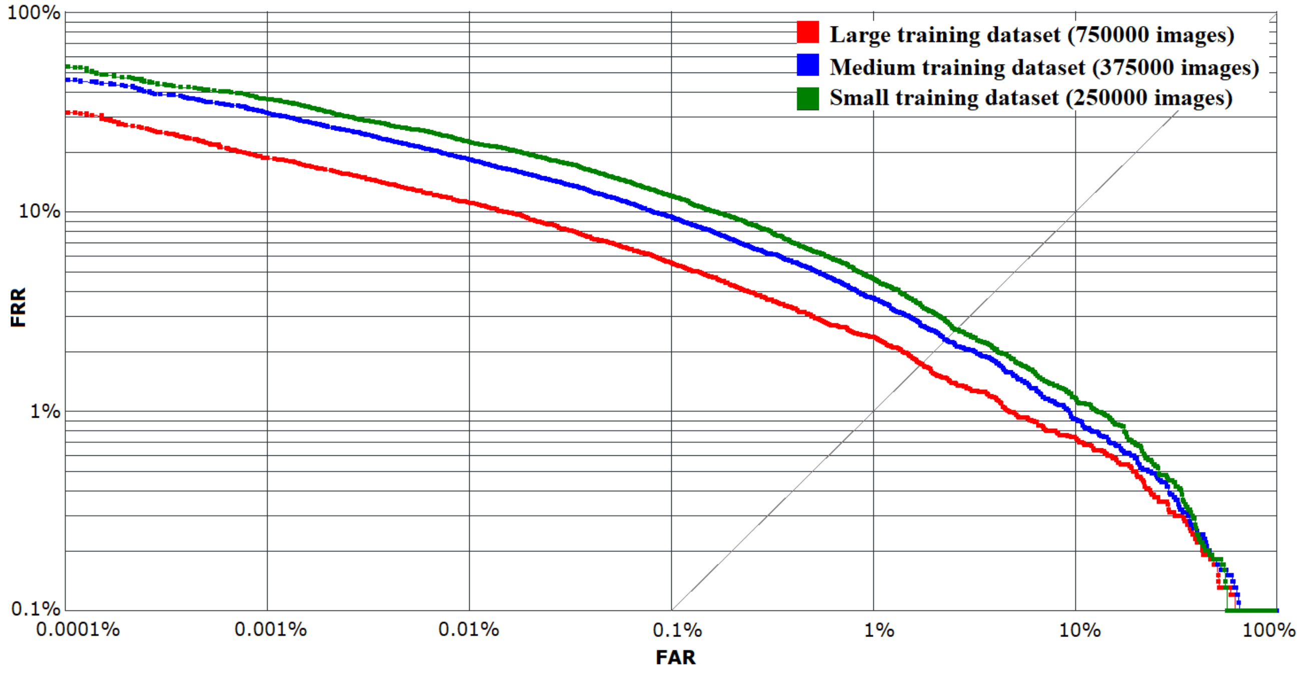

Figure 6. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with cross-entropy loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

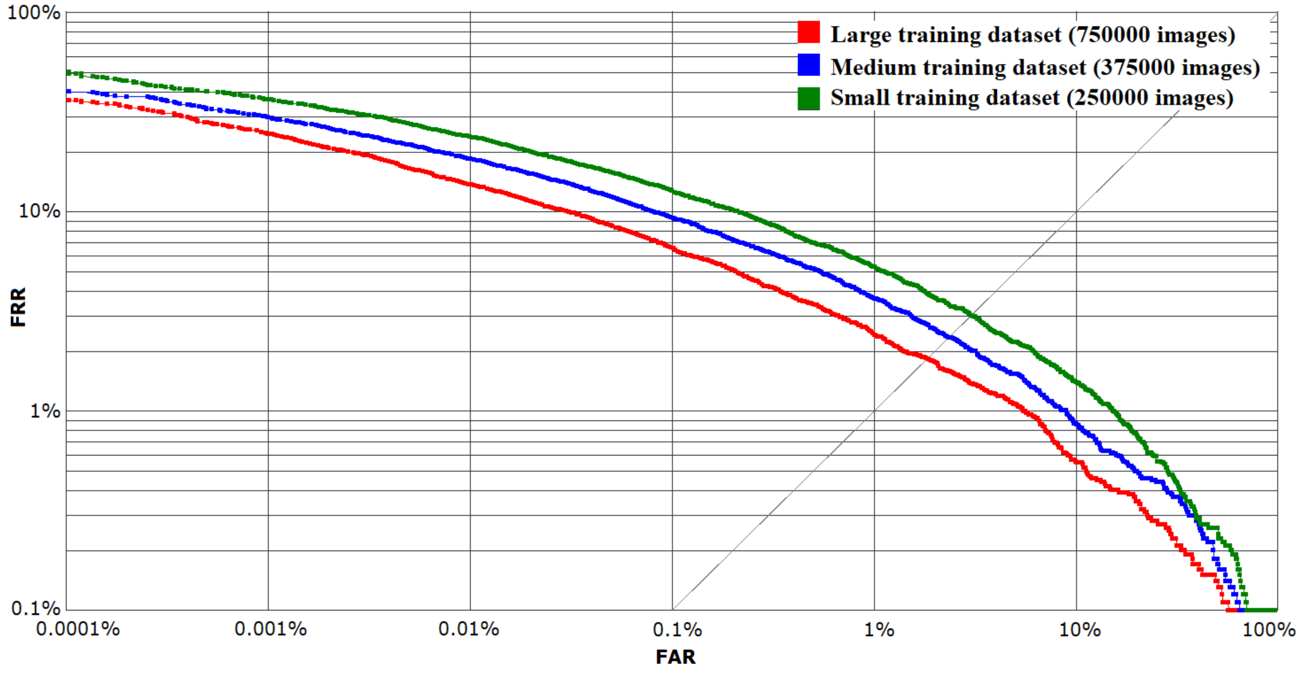

Figure 7. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with triplet loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

In Figure 8 (full-sized view can be found in Appendix 2: Figure 31) all three experiments with the additive angular margin loss are presented for MS-Celeb-1M test set. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 29, 30. It can be noticed that the additive angular margin loss demonstrates behavior opposite to the triplet loss. Enlargement of training dataset improves the lowest FAR DET curve's region. Moving towards the EER point this improvement is monotonically decreasing. Such behavior can be beneficial in tasks like MEDS-II and MS-Celeb-1M face verification where evaluation is performed using FRR value at 0.001% and 0.01% FAR points. By comparing plots between different datasets, it can be stated that results agree between themselves.

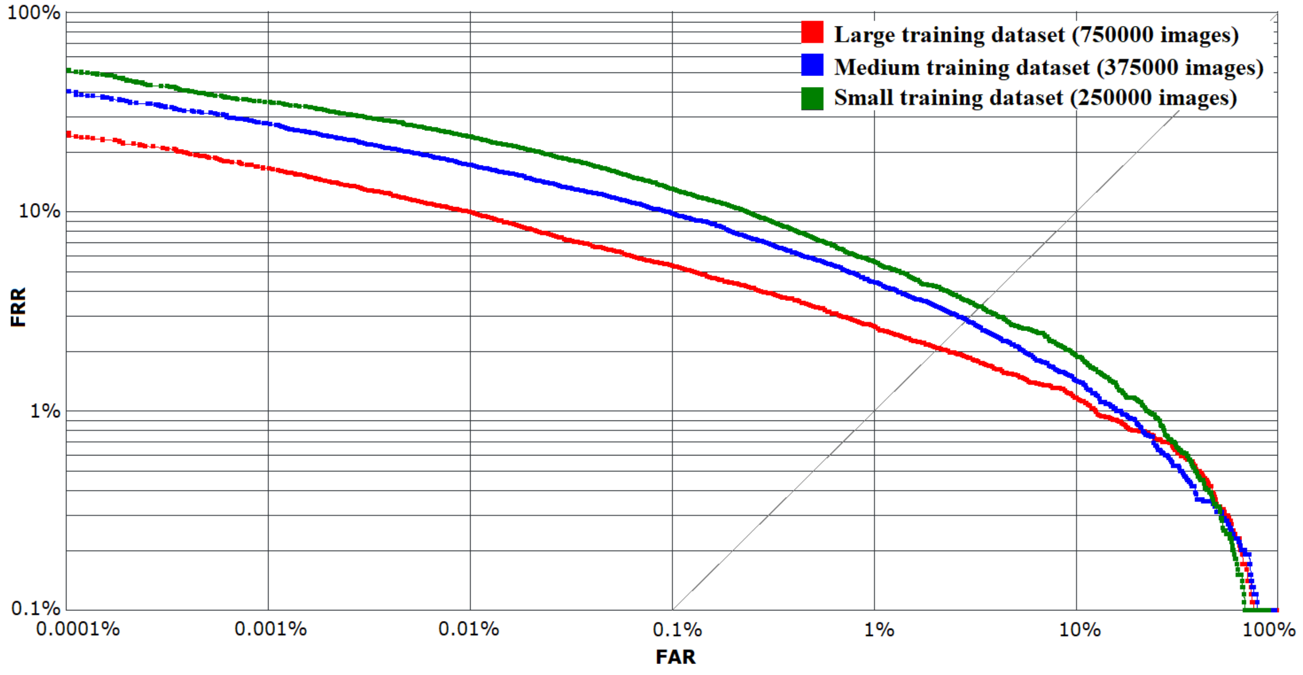

Figure 8. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

3.6. Triplet loss fine-tuning

The next set of experiments was additional triplet loss fine-tuning of models trained with cross-entropy and additive angular margin losses from the first set of experiments. In [LMJ+17] authors stated that models trained with the triplet loss can converge to a suboptimal local minimum due to difficulties arising while mining appropriate training triplets. To overcome this issue, they pre-trained model with more stable loss and subsequently fine-tuned it with triplet loss. Pre-training stage can be considered as incorporation of some kind of prior knowledge about the task into model parameters initialization. Two trained models taken from the first set of experiments were used as a base for the triplet loss fine-tuning stage. The same results evaluation protocol was applied as in corresponding triplet loss training from scratch experiments.

For the first experiment, the cross-entropy loss function was chosen. The ResNet21 model trained in the first set of experiments was additionally trained with the triplet loss. Considering that the fine-tuning process generally requires much lower learning rates, it was decided to set this hyperparameter to 0.001 value. Rest of hyperparameters settings were copied from the triplet loss experiment described in subsection 3.2. The complete experiment required 243 training epochs and took around 64 hours. DET curves for MS-Celeb-1M test dataset are presented in Figure 9 (full-sized view can be found in Appendix 2: Figure 36). The red curve represents the original model trained with the cross-entropy loss and the blue curve represents the model that was additionally fine-tuned with the triplet loss. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 34, 35.

The second experiment was performed with a model trained using additive angular margin loss. The same training process protocol was applied as in the first experiment. The complete experiment required 27 training epochs and took around 8 hours. DET curves for MS-Celeb-1M test dataset are presented in Figure 10 (full-sized view can be found in Appendix 2: Figure 39). The red curve represents the original model trained with the additive angular margin loss and the blue curve represents the model that was additionally fine-tuned with the triplet loss. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 37, 38.

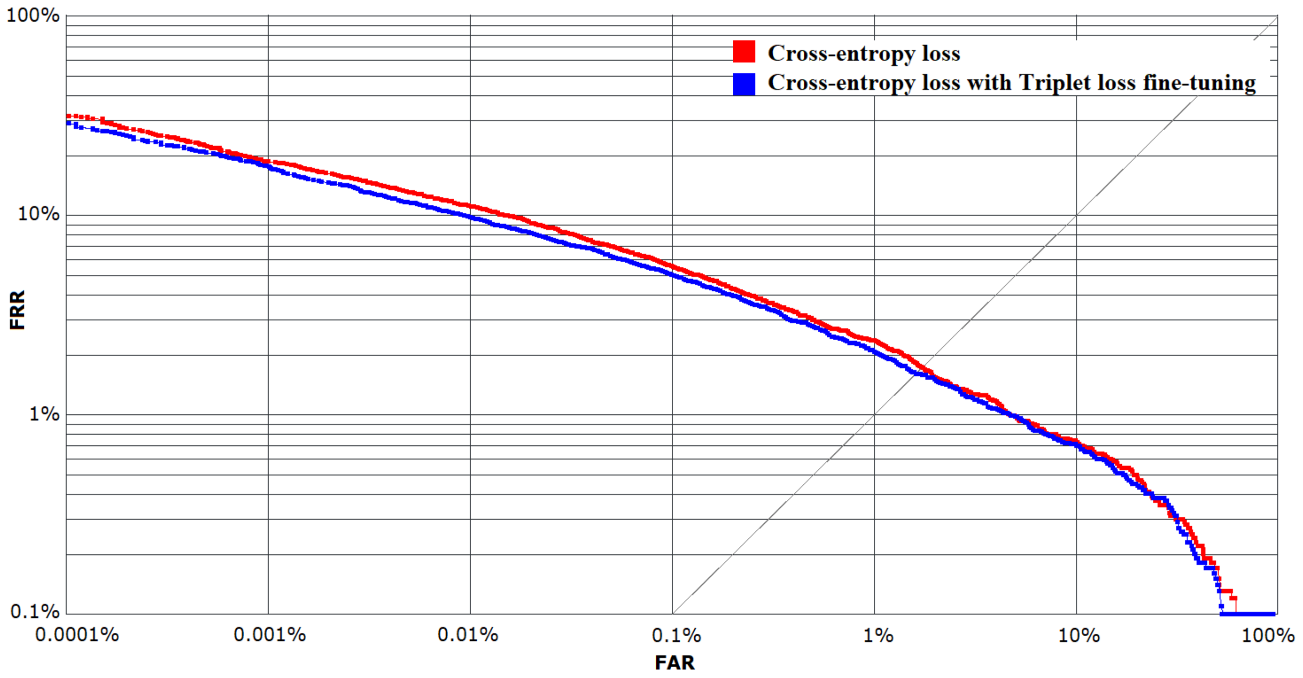

Figure 9. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with cross-entropy loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

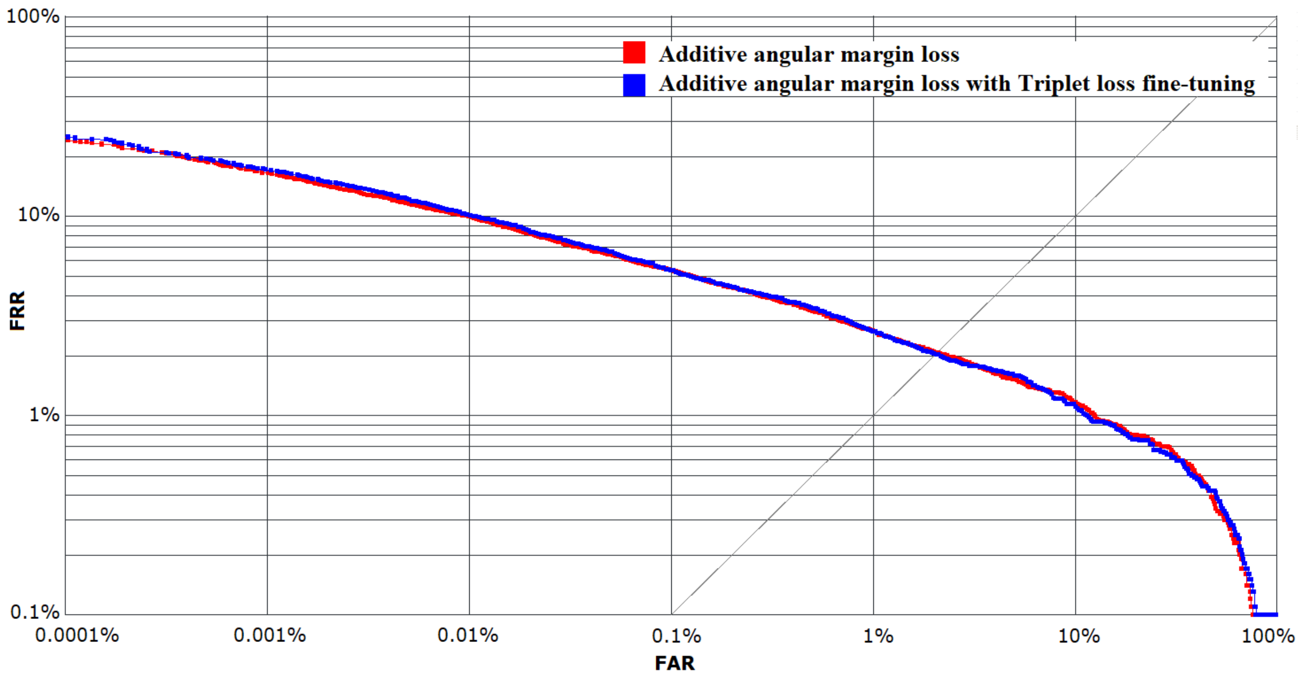

Figure 10. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

By analyzing DET curves in Figure 9, it can be noticed that additional triplet loss fine-tuning of models pre-trained with the cross-entropy loss can bring slight performance improvements. Considering the results presented in subsection 3.2, this can be beneficial for tasks like LFW face verification that are evaluated using the EER point of a DET curve. Results of the cross-entropy loss, which is the best performing loss at this point, can be outperformed by additional triplet loss fine-tuning stage. However, considering the low FAR region, the results of the model trained with the additive angular margin loss alone still cannot be surpassed. Therefore, for tasks like MEDS-II and MS-Celeb-1M face verifications, it is still more useful to select additive angular margin loss.

By analyzing DET curves from Figure 10, it can be stated that the application of the triplet loss fine-tuning to models pre-trained with the additive angular margin loss does not bring any improvements. One of the possible reasons for such behavior is that these two losses optimize DET curve’s regions that are too far away from one another to be synergistically combined.

By comparing plots between different datasets in Appendix 2: Figures 34–39, it can be stated that results agree between themselves. Final results, evaluated according to described earlier dataset protocols, are presented in Table 4.

Table 4. Evaluation protocol results: ResNet21 CNN trained with base loss and additionally fine-tuned with triplet loss using MS-Celeb-1M large dataset.

	Cross-entropy loss with triplet loss fine-tuning	**Additive angular margin loss with triplet loss fine-tuning**
LFW (EER)	0.45%	0.6667%
MEDS-II (FRR@FAR=0.001%)	4.263%	3.81%
MS-Celeb-1M (FRR@FAR=0.01%)	9.91%	10.17%

3.7. ResNet13 architecture training

The last set of experiments was done using smaller ResNet13 CNN architecture and MS-Celeb-1M large dataset, consisting of 750000 images of 15000 individuals. The main objective of these experiments is to investigate dependencies between loss functions and model depth. The same training process and results evaluation protocols were applied as in corresponding experiments with ResNet21 architecture.

The first experiment was conducted with ResNet13 architecture and the cross-entropy loss function. The training was performed with the same settings of SGD optimizer's hyperparameters as in the first set of experiments. Using the information provided by an early stopping criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 303 training epochs and took around 72 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 11 in blue color (full-sized view can be found in Appendix 2: Figure 42). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 40, 41.

The second experiment was performed with the triplet loss. Hyperparameters settings were copied from the corresponding experiment with ResNet21 model. The training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 876 training epochs and took around 201 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 11 in green color (full-sized view can be found in Appendix 2: Figure 42). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 40, 41.

The last experiment was ResNet13 architecture training with the additive angular margin loss. Hyperparameters settings were copied from the corresponding experiment with ResNet21 model. Using the information provided by an early stopping criterion, the training process was restarted twice with learning rate lowered by a factor of ten (0.01 and 0.001). The complete experiment required 324 training epochs and took around 68 hours. DET curve for MS-Celeb-1M test dataset is presented in Figure 11 in red color (full-sized view can be found in Appendix 2: Figure 42). Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 40, 41.

By analyzing DET curves, it can be stated that loss functions behavior pattern is not changed with model depth decrease. All observations from the first set of experiments with ResNet21 architecture are applicable to the current set of experiments. By comparing plots between different datasets, it can be stated that results agree between themselves. Final results, evaluated according to described earlier dataset protocols, are presented in Table 5.

Table 5. Evaluation protocol results: ResNet13 CNN trained with MS-Celeb-1M large dataset.

	Cross-entropy loss	**Triplet loss**	**Additive angular margin loss**
LFW (EER)	0.5333%	0.6333%	0.6167%
MEDS-II (FRR@FAR=0.001%)	3.311%	6.304%	2.721%
MS-Celeb-1M (FRR@FAR=0.01%)	12.31%	15.36%	10.19%

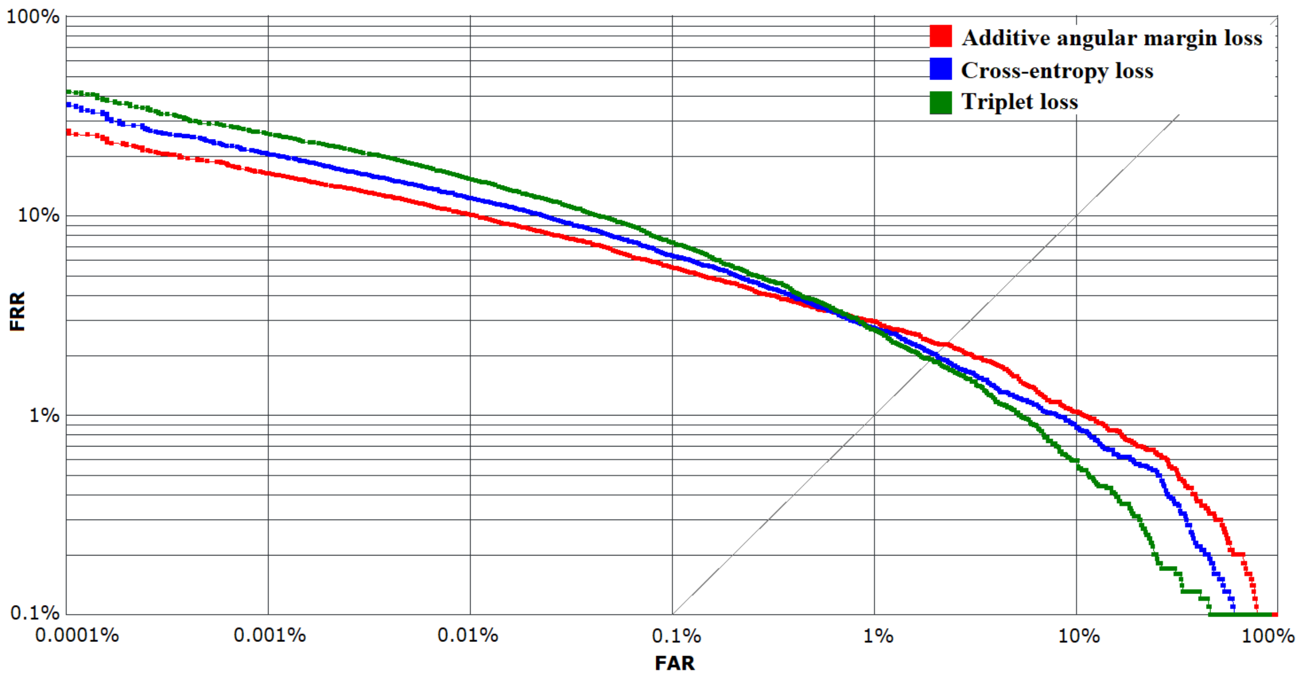

Figure 11. MS-Celeb-1M test set DET curves of ResNet13 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

3.8. Model depth impact

In this section, results from the first and the last experiment sets are aggregated by CNN architecture and presented for analysis of model depth impact. In Figure 12 (full-sized view can be found in Appendix 2: Figure 45) experiments with ResNet21 and ResNet13 architectures trained using the cross-entropy loss are provided for MS-Celeb-1M test set. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 43, 44. By analyzing DET curves it can be noticed that the cross-entropy loss can effectively utilize additional network's capacity and to improve results at all curve's points. The largest improvement is at the ERR point, where the cross-entropy loss is performing the best in the majority of experiments so far. Some minor discrepancies can be seen for MEDS-II dataset experiments in Appendix 2: Figure 44 in the zero FAR region. However, considering that MEDS-II dataset has small positive class pairs number and that DET curve is a cumulative performance measure, it is quite probable that only couple additionally misclassified impostor pairs was enough to create this hump.

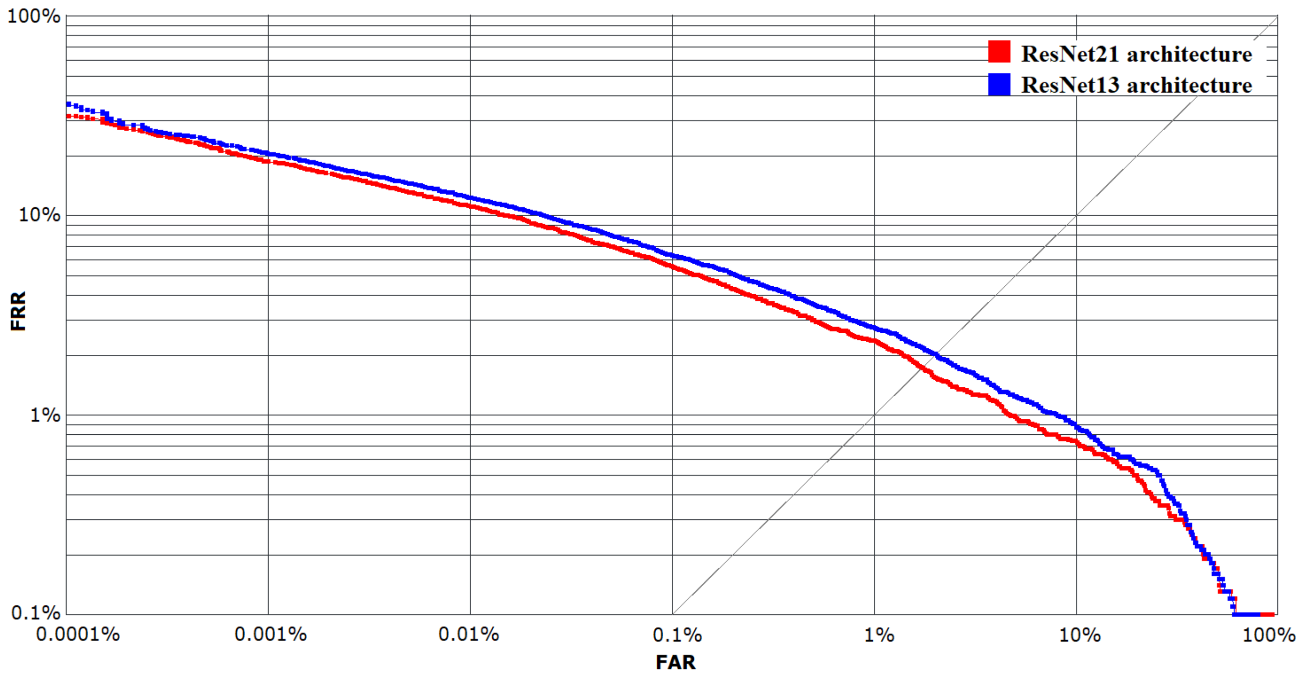

Figure 12. MS-Celeb-1M test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with cross-entropy loss using MS-Celeb-1M large dataset.

In Figure 13 (full-sized view can be found in Appendix 2: Figure 48) experiments with ResNet21 and ResNet13 architectures trained using the triplet loss are presented for MS-Celeb-1M test set. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 46, 47. As can be seen, the triplet loss benefits less from additional network capacity than the cross-entropy loss. The largest improvement can be noticed in the low FAR region, where it performs the worst out of three studied losses. By comparing plots between different datasets, it can be stated that results agree between themselves.

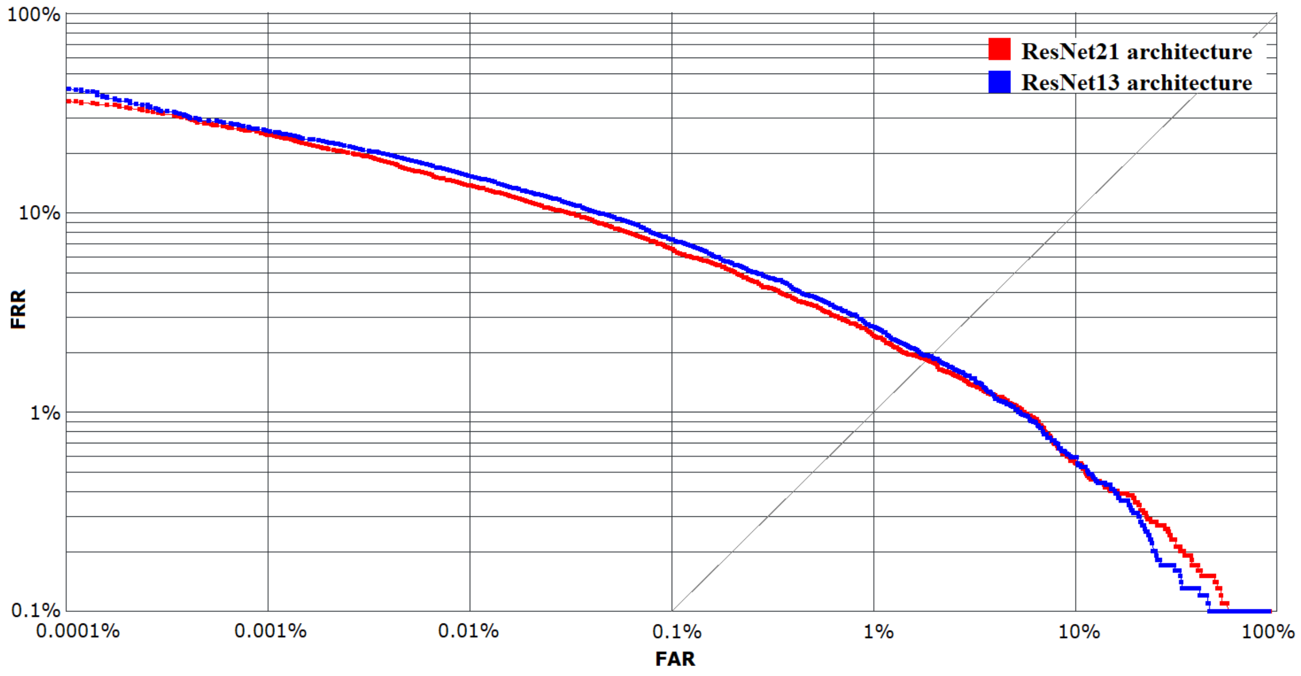

Figure 13. MS-Celeb-1M test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with triplet loss using MS-Celeb-1M large dataset.

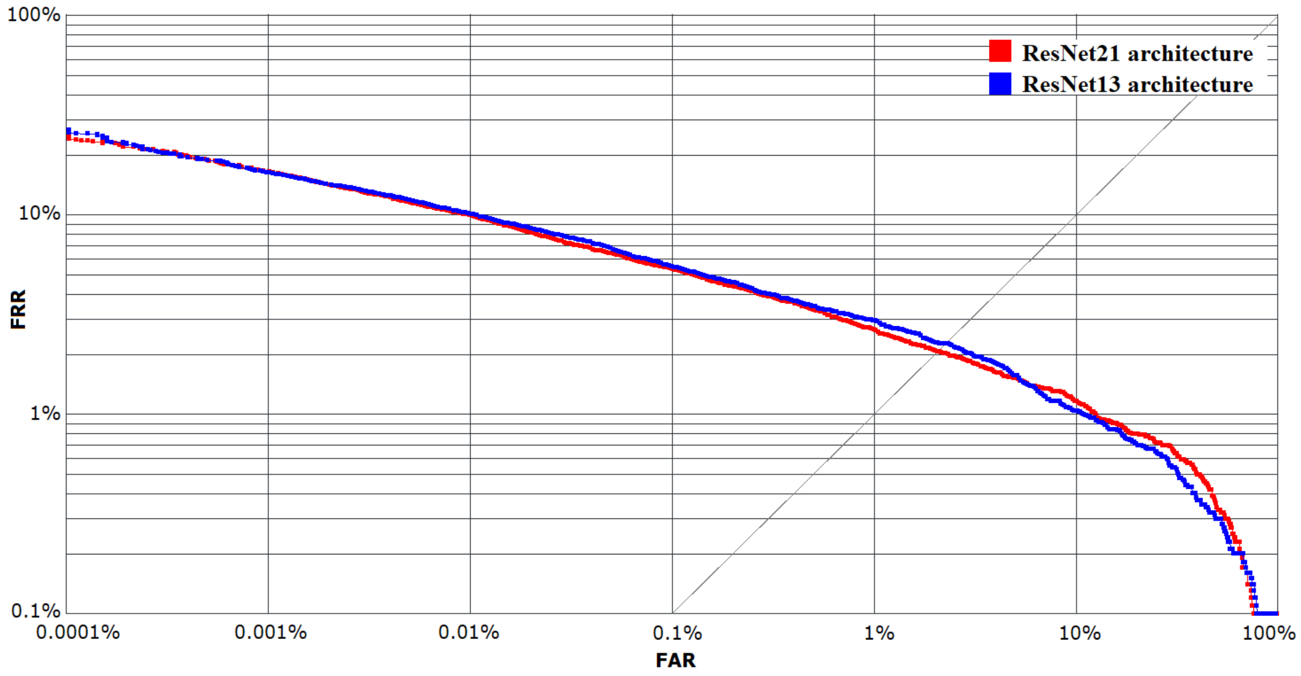

Figure 14. MS-Celeb-1M test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with additive angular margin loss using MS-Celeb-1M large dataset.

In Figure 14 (full-sized view can be found in Appendix 2: Figure 51) experiments with ResNet21 and ResNet13 architectures trained using the additive angular margin loss are presented for MS-Celeb-1M test set. Test DET curves for LFW and MEDS-II datasets are provided in Appendix 2: Figures 49, 50. By analyzing DET curves, it can be seen that the additive angular margin loss mostly does not benefit from additional network capacity. More thorough results analysis from MEDS-II test dataset in Appendix 2: Figure 50 makes it possible to detect probable ResNet21 model overfitting. Therefore, it can be stated that this loss requires much smaller network architectures to achieve competitive performance and can additionally benefit from training dataset enlargement.

3.9. Results generalization

In the last chapter results from all conducted experiments are collected and generalized by comparative analysis using criterions described at the beginning of this part. Obtained results show that different deep learning loss functions are represented by different behavior patterns on a DET or ROC curve.

Biometric face verification tasks that require optimization of the low FAR DET curve's region, like MEDS-II and MS-Celeb-1M benchmarks or biometric security systems at airports, also require substantial training datasets. In case of having not sufficiently large training dataset, all studied losses perform similarly in the low FAR region. In such settings, probably, the cross-entropy loss is the best choice as it has a better area under the curve (AUC). In cases when a substantial dataset is available, it is more beneficial to use the additive angular margin loss – it can be seen by analyzing experiment sets with large and medium MS-Celeb-1M training datasets. However, no particular research on how to find optimal dataset size was performed during this study. In general, optimizing FRR value at the low FAR region is common not only for biometric classification. Medical imaging tasks where misclassified malignant samples are much more undesirable than misclassified benign samples also have similar evaluation protocols.

For face verification tasks that require optimization of the EER point of the DET curve, like LFW benchmark or various non-critical systems such as biometric time attendance control, it is more beneficial to use the cross-entropy loss. In all conducted experiments models trained with the cross-entropy loss produced the best results at the EER point. In cases when a substantial dataset is available, the triplet loss can be used as well, as it produces comparable results – it can be seen by analyzing experiment sets with large and medium MS-Celeb-1M training datasets. In general, optimizing EER is a natural choice for binary classification tasks without specific requirements to a curve shape, as it produces more unbiased classifier.

In general, considering facts that the triplet loss requires much longer training time and that it is always possible to find another loss function performing at the same level or even better for a selected particular curve's region, it can be concluded that this loss is not well-suited for training face verification classifiers from scratch. Most probably, this observation can be transferred to generic binary classification problems, as loss function's rate of convergence is roughly constant across different tasks. However, it still can be applied for fine-tuning models pre-trained with the cross-entropy loss in biometric tasks. In such settings, the long training time issue can be partially mitigated and the strongest aspects of the triplet loss function (optimization of the EER point) potentially can be transferred to a pre-trained model. Unfortunately, the triplet loss fine-tuning technique does not bring any improvements while applied to models trained with the additive angular margin loss. A hypothesis explaining such behavior was proposed – it is possible to productively combine loss functions that optimize close DET or ROC curve's regions only.

Dataset size increase brings improvement for all studied losses in all conducted experiments. However, improvements are not uniform for all points of the metric's curve. By analyzing produced DET curves in Appendix 2: Figures 25–33, it can be noticed that each loss function improves its strongest optimization region bit more than other points. The additive angular margin loss uses extra training samples to improve more a low FAR region, while cross-entropy and triplet losses prefer the EER point. To sum up, it is usually worth to invest more resources for additional training data acquisition if it is possible.

Unfortunately, simple model depth increase not obviously brings test set performance improvements for all studied losses. Considering biometric face verification problems, the cross-entropy loss benefits the most from the additional model's capacity. This fact justifies the hypothesis that it is possible to improve further cross-entropy results in this study without enlarging training dataset. The triplet loss shows limited improvement with model depth increase. This means that further model size growth probably will not bring any additional benefits. While analyzing DET curves produced by the additive angular margin loss experiments, it was noticed that the model's depth increase probably caused overfitting. Reset13 architecture produces the same or even better results than the baseline ResNet21 architecture for all test sets used in this study. This observation means that the additive angular margin loss requires much smaller network architectures to achieve competitive performance. In general, a neural network's size increase usually should be done together with training dataset enlargement to be able to obtain the maximum performance gain.

Having sufficiently large datasets, training time requirements for cross-entropy and additive angular margin losses are quite similar for face verification tasks. However, with smaller datasets, models trained with the additive angular margin loss converge faster. The triplet loss, as was stated earlier, requires much longer time in all training process settings.

Conclusion

In this master thesis dependencies between deep learning loss function form and its ROC or DET curve’s strongest optimization region or point were investigated for biometric face verification problems. The main goal of this analysis is to help to select appropriate of the shelf loss function under different binary classification evaluation metric constraints.

Three, currently the most popular, deep learning loss functions for binary classification in face verification tasks were explored: cross-entropy loss, triplet loss, and additive angular margin loss.

Thesis conclusions:

1. The additive angular margin loss function produces the best results in face verification tasks that are evaluated using low FAR DET curve’s region. Biometric security passport gates at airports can be considered as an example of such tasks. To achieve superior results it requires a substantial amount of training data and can be used with smaller neural network architectures.
2. It worth to use the cross-entropy loss function for face verification problems that are evaluated by the EER DET curve’s point and do not have specific preferences for error types. In addition, this loss function produces the best results in training data shortage settings. It worth to use deeper neural network architectures with the cross-entropy loss function in order to obtain better performance.
3. Considering training time requirements of the triplet loss function, it worth to use it only for fine-tuning models pre-trained with the cross-entropy loss function. It optimizes the most the EER DET curve’s point and is applicable to the same face verification tasks as the cross-entropy loss.

As a research output, advice pack how to choose appropriate loss function for differently constrained binary classification problems from biometric face verification domain was compiled.

References

[CM04] C. Cortes, M. Mohri. AUC Optimization vs. Error Rate Minimization. Proceedings of the 16th International Conference on Neural Information Processing Systems, 2004, pp. 313–320.

[DGX+18] J. Deng, J. Guo, N. Xue, S. Zafeiriou. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. arXiv preprint, 2018, pp. 1–11.

[DNH+18] E. Dodds, H. Nguyen, S. Herdade, J. Culpepper, A. Kae, P. Garrigues. Learning Embeddings for Product Visual Search with Triplet Loss and Online Sampling. arXiv preprint, 2018, pp. 1–6.

[DY14] L. Deng, D. Yu. Deep Learning: Methods and Applications. Foundations and Trends® in Signal Processing, 7(3-4), 2014, pp. 197–387.

[ESM+17] E. Eban, M. Schain, A. Mackey, A. Gordon, R.A. Saurous, G. Elidan. Scalable Learning of Non-Decomposable Objectives. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS), 54, 2017, pp. 832–840.

[FOW+11] A.P. Founds, N. Orlans, G. Whiddon. Multiple Encounter Dataset II (Deceased Persons) MEDS-II: Data Description Document. Technical Report MTR100439, 2011, pp. 1–24.

[GBB11] X. Glorot, A. Bordes, Y. Bengio. Deep Sparse Rectifier Neural Networks. Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS), 15, 2011, pp. 315–323.

[GBC16] I. Goodfellow, Y. Bengio, A. Courville. Deep Learning. MIT Press, 2016, 785 pages.

[GZH+16] Y. Guo, L. Zhang, Y. Hu, X. He, J. Gao. MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition. Proceedings of the 14th European Conference on Computer Vision (ECCV), Springer International Publishing AG, 2016, pp. 87–102.

[HSK+12] G. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint, 2012, pp. 1–18.

[HRB+07] G.B. Huang, M. Ramesh, T. Berg, E. Learned-Miller. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. University of Massachusetts, Amherst, Technical Report 07-49, 2007, pp 1–11.

[HZR+15] K. He, X. Zhang, S. Ren, J. Sun. Delving Deep into Rectifiers: Surpassing Human-

Level Performance on ImageNet Classification. Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1026–1034.

[HZR+16] K. He, X. Zhang, S. Ren, J. Sun. Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778.

[IS15] S. Ioffe, C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning, JMLR: W&CP Volume 37, 2015, pp. 448–456.

[KSH12] A. Krizhevsky, I. Sutskever, G.E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. Advances in Neural Information Processing Systems, 25, 2012, pp. 1097–1105.

[KTT+12] Y. Kim, K.A. Toh, A.B.J. Teoh, H.L. Eng, W.Y. Yau. An online AUC formulation for binary classification. Pattern Recognition, 45(6), 2012, pp. 2266–2279.

[LKF10] Y. LeCun, K. Kavukcuoglu, C. Farabet. Convolutional networks and applications in vision. Proceedings of 2010 IEEE International Symposium on Circuits and Systems, 2010, pp. 253–256.

[LMJ+17] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, Z. Zhu. Deep Speaker: an End-to-End Neural Speaker Embedding System. arXiv preprint, 2017, pp. 1–8.

[MDC+02] M.C. Mozer, R. Dodier, M.D. Colagrosso, C. Guerra-Salcedo, R. Wolniewicz. Prodding the ROC curve: Constrained optimization of classifier performance. Advances in Neural Information Processing Systems, 14(1-2), 2002, pp. 1409–1415.

[NIST19] National Institute of Standards and Technology. Ongoing Face Recognition Vendor Test (FRVT), Part 1: Verification, 2019/04/12.
[accessed: 2019-04-24]. Internet address: <https://www.nist.gov/sites/default/files/documents/2019/04/15/frvt_report_2019_04_12.pdf>

[PGC+17] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer. Automatic differentiation in PyTorch. 30th Conference on Neural Information Processing Systems (NIPS) workshop, 2017, pp. 1–4.

[Pre12] L. Prechelt. Early Stopping – but when? Neural Networks: Tricks of the Trade, 7700, Springer, 2012, pp 53–67.

[RPD+16] O. Rippel, M. Paluri, P. Dollar, L. Bourdev. Metric Learning with Adaptive Density Discrimination. International Conference on Learning Representations (ICLR), 2016.

[SKP15] F. Schroff, D. Kalenichenko, J. Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 815–823.

[SSZ+16] Y. Song, A.G. Schwing, R.S. Zemel, R. Urtasun. Training Deep Neural Networks via Direct Loss Minimization. Proceedings of the 33rd International Conference on Machine Learning, 48, 2016, pp. 2169–2177.

[STB+17] A.A.A Setio, A. Traverso, T. de Bel, M.S.N. Berens, C. van den Bogaard, P. Cerello, H. Chen, Q. Dou, M.E. Fantacci, B. Geurts, R. van der Gugten, P.A. Heng, B. Jansen, M.M.J. de Kaste, V. Kotov, J.Y. Lin, J.T.M.C. Manders, A. Sonora-Mengana, J.C. Garcıa-Naranjo, E. Papavasileiou, M. Prokop, M. Saletta, C.M. Schaefer-Prokop, E.T. Scholten, L. Scholten, M.M. Snoeren, E.L. Torres, J. Vandemeulebroucke, N. Walasek, G.C.A. Zuidhof, B. van Ginneken, C. Jacobs. Validation, comparison, and combination of algorithms for automatic detection of pulmonary nodules in computed tomography images: The LUNA16 challenge. Medical Image Analysis, 42, 2017, pp. 1–13.

[VWB16] A. Veit, M.J. Wilber, S. Belongie. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. 29th Conference on Neural Information Processing Systems (NIPS), 2016, pp. 550–558.

[WD18] M. Wang, W. Deng. Deep Face Recognition: A Survey. arXiv preprint, 2018, pp. 1–24.

[ZF14] M.D. Zeiler, R. Fergus. Visualizing and Understanding Convolutional Networks. Computer Vision – ECCV 2014, 8689, Springer International Publishing, 2014, pp. 818–833.

Appendices

Appendix 1. ResNet CNN architecture's blocks structure

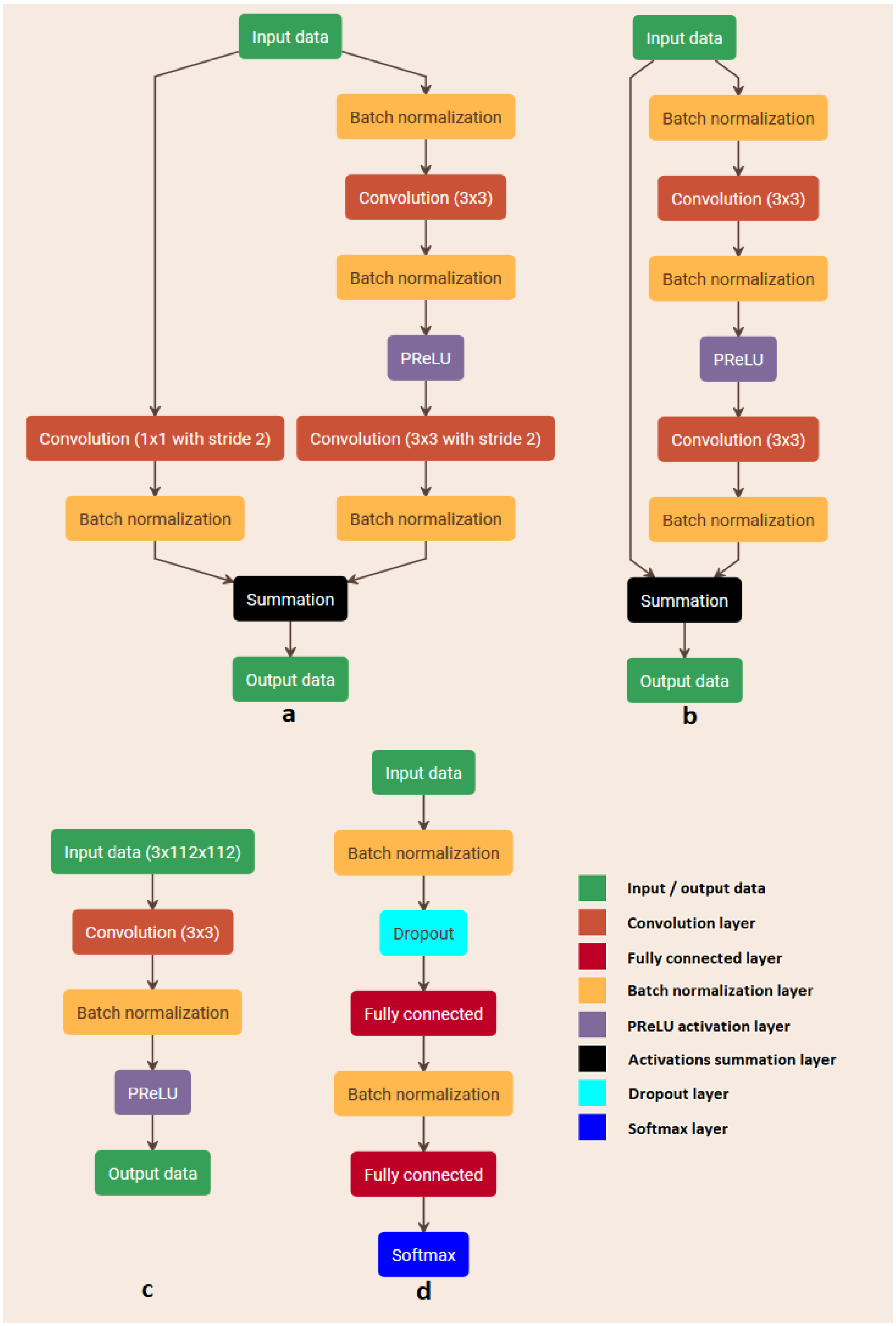

Figure 15. ResNet CNN architecture's blocks structure: a) Residual block with spatial dimension reduction, b) Ordinary residual block, c) Network input block, d) Network output block.

Appendix 2. Detection error tradeoff curves

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.65%	3.067%	3.067%	3.067%	3.067%
6000	3000	3000	0.5%	1.633%	1.633%	1.633%	1.633%
6000	3000	3000	0.6%	2.167%	2.167%	2.167%	2.167%

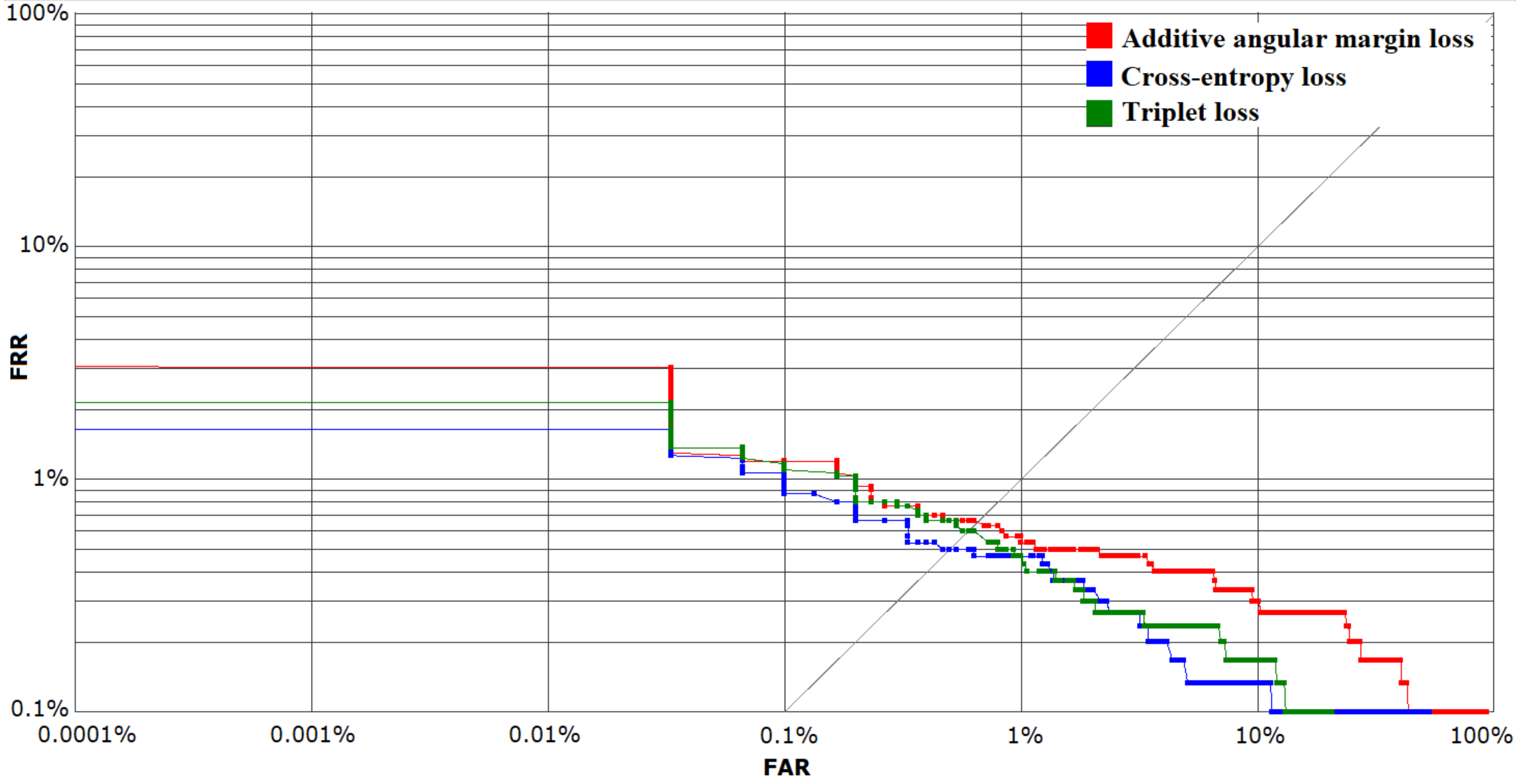

Figure 16. LFW test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8084%	9.478%	9.478%	3.356%	1.769%
1477440	4410	1473030	0.8126%	14.51%	14.51%	4.626%	1.769%
1477440	4410	1473030	0.8124%	15.83%	15.83%	5.397%	2.721%

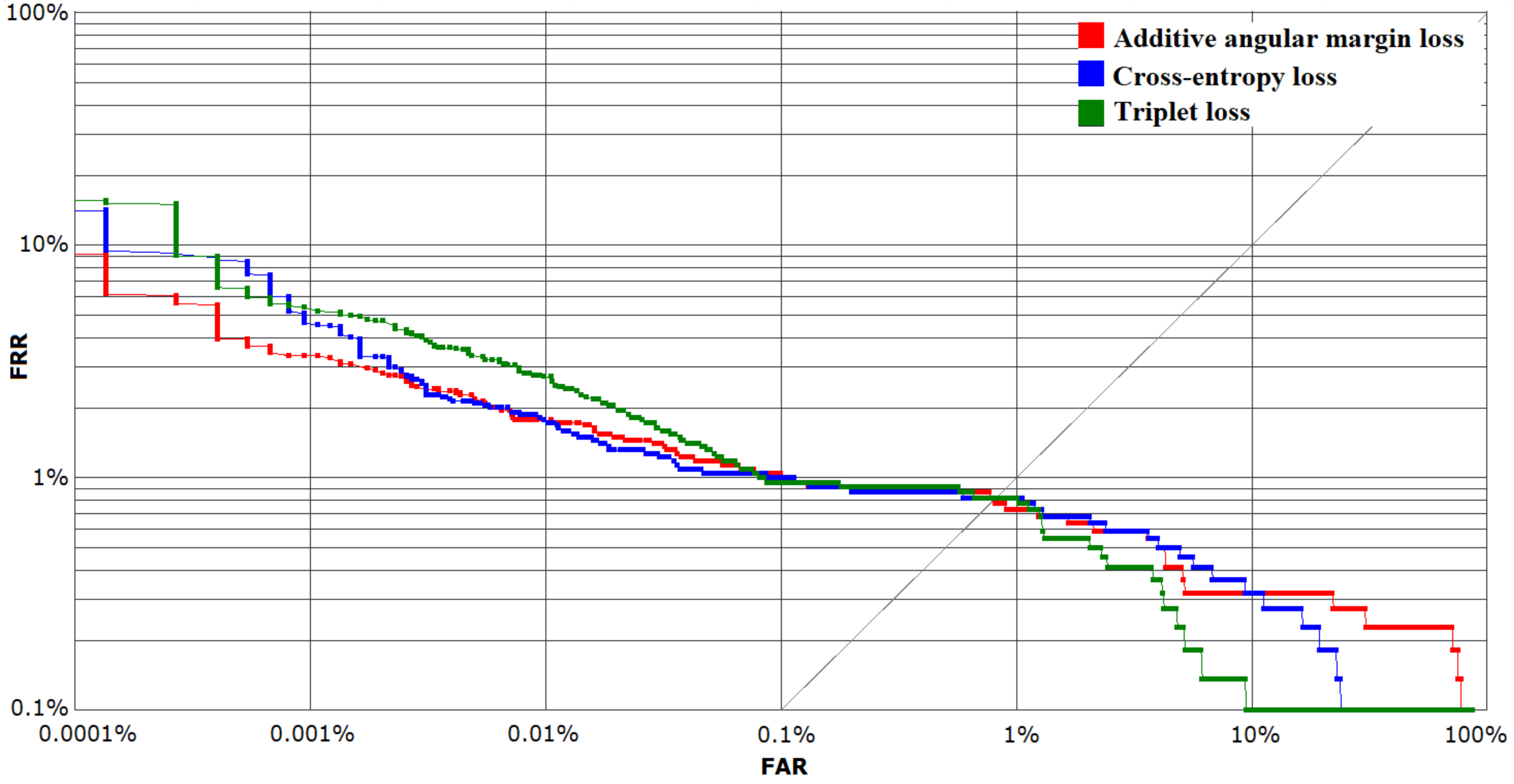

Figure 17. MEDS-II test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	2.065%	96.58%	25.04%	16.58%	9.99%
24995000	20000	24975000	1.732%	97.3%	31.75%	18.75%	11.18%
24995000	20000	24975000	1.831%	98.39%	36.62%	24.92%	13.76%

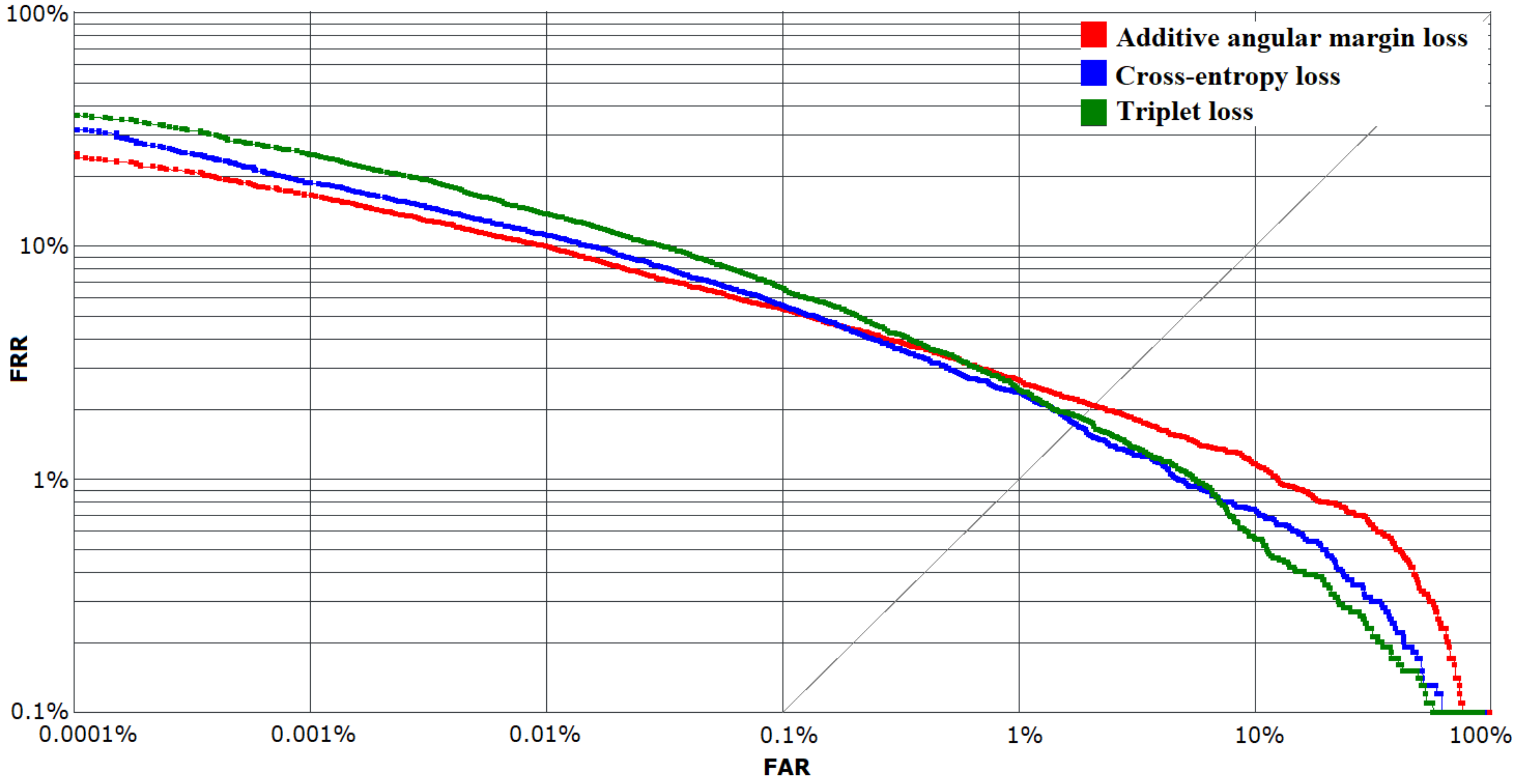

Figure 18. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.9667%	4.333%	4.333%	4.333%	4.333%
6000	3000	3000	0.75%	6.8%	6.8%	6.8%	6.8%
6000	3000	3000	0.8333%	12.57%	12.57%	12.57%	12.57%

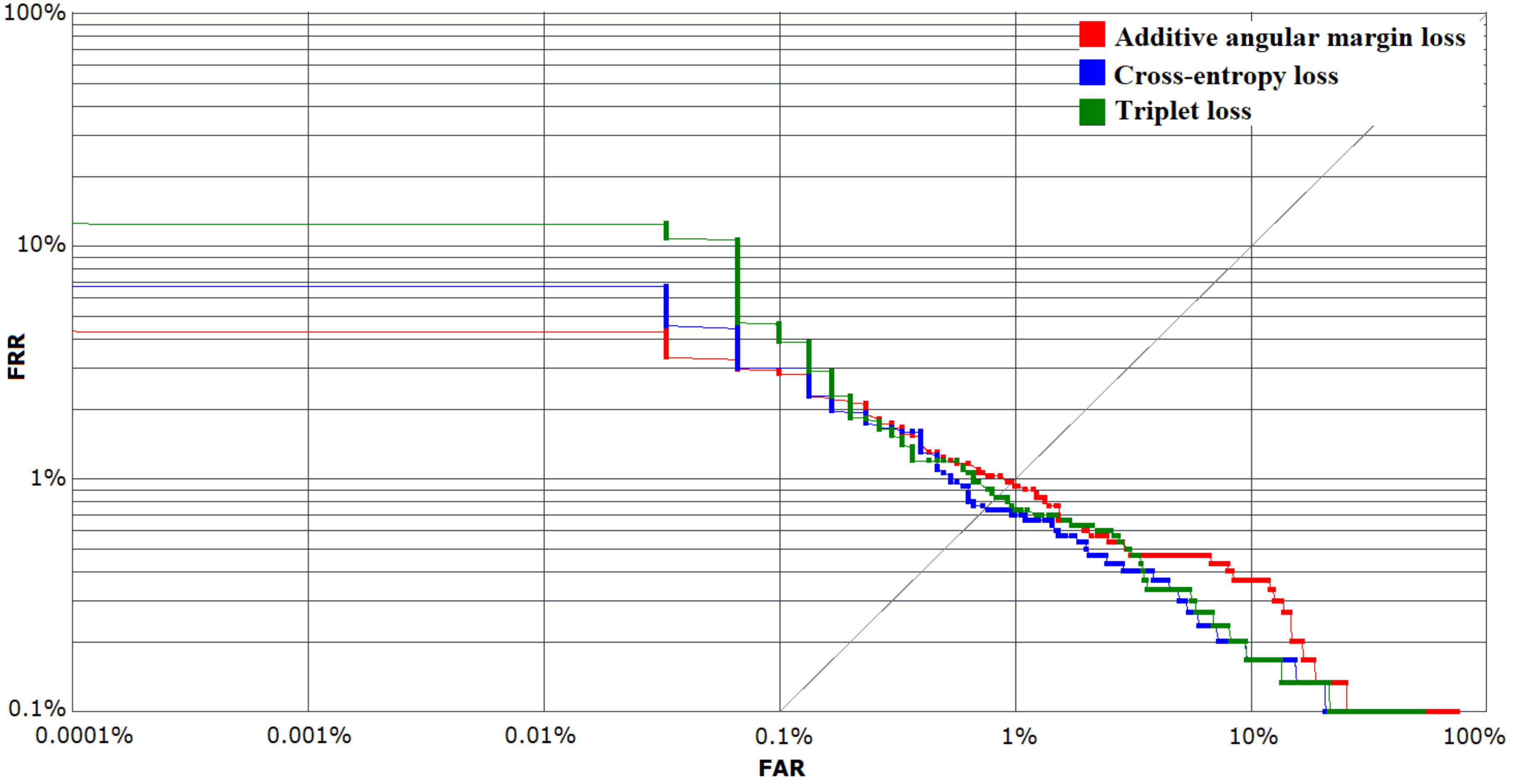

Figure 19. LFW test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M medium dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.9043%	11.25%	11.25%	7.347%	3.9%
1477440	4410	1473030	0.8162%	21.32%	21.32%	13.74%	5.986%
1477440	4410	1473030	0.8526%	17.87%	17.87%	8.254%	4.58%

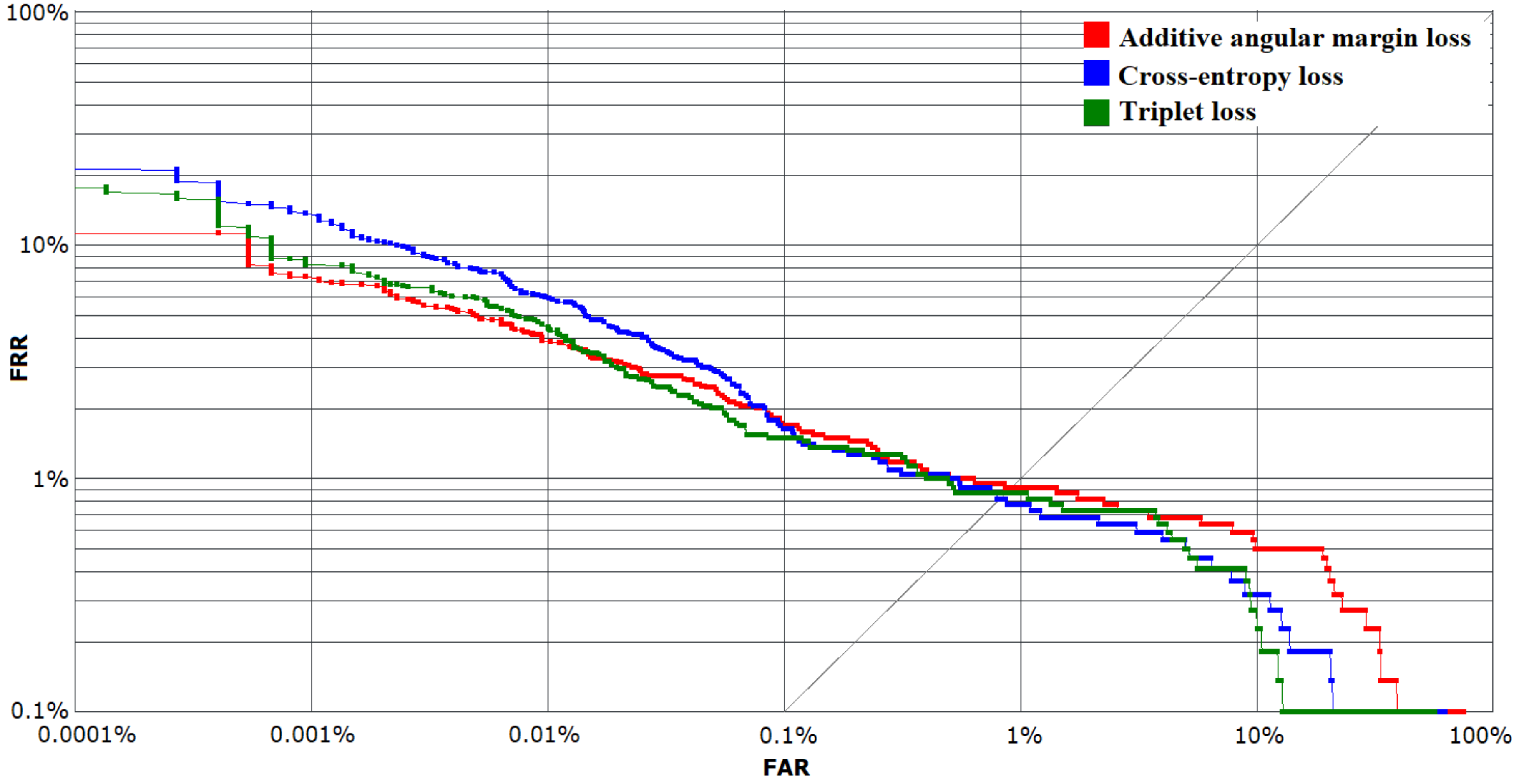

Figure 20. MEDS-II test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M medium dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	2.853%	98.56%	40.56%	27.69%	17.24%
24995000	20000	24975000	2.278%	98.33%	46.12%	31.66%	18.43%
24995000	20000	24975000	2.327%	98.83%	40.48%	30.08%	18.56%

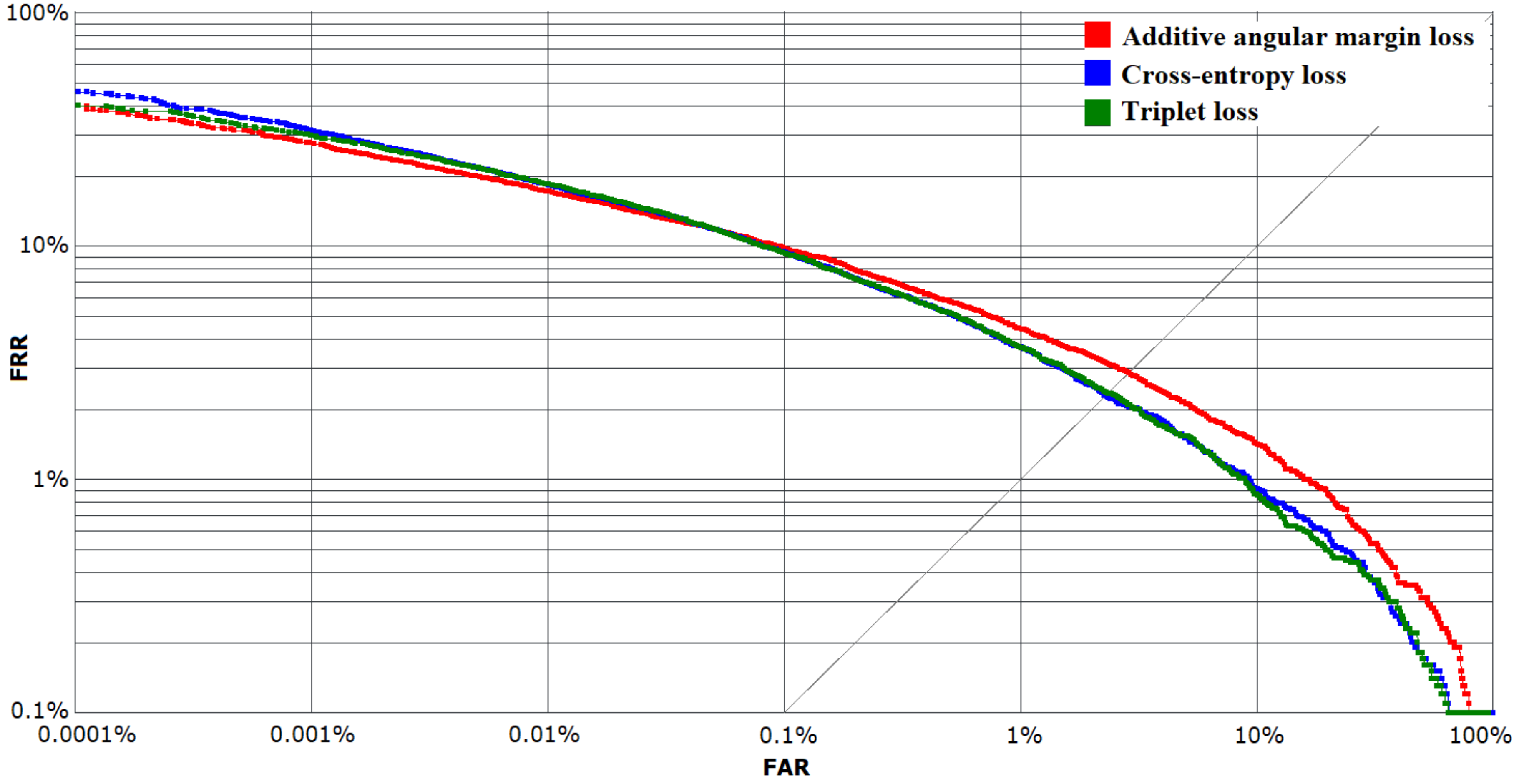

Figure 21. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M medium dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	1.25%	9.133%	9.133%	9.133%	9.133%
6000	3000	3000	1.033%	8.767%	8.767%	8.767%	8.767%
6000	3000	3000	1.233%	8.2%	8.2%	8.2%	8.2%

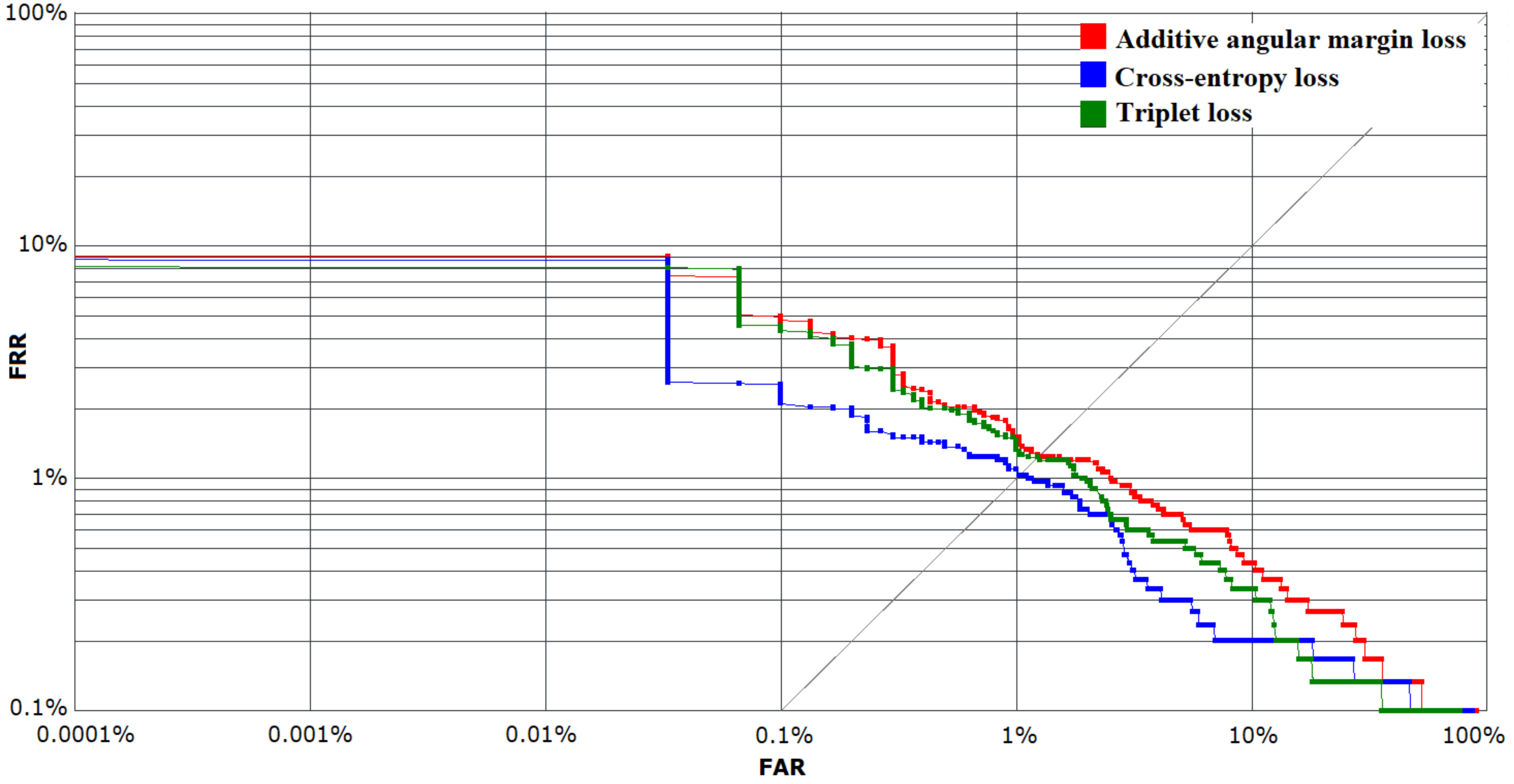

Figure 22. LFW test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M small dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	1.214%	31.38%	31.38%	13.38%	6.893%
1477440	4410	1473030	0.7948%	22.77%	22.77%	12.29%	6.349%
1477440	4410	1473030	1.086%	16.69%	16.69%	12.47%	7.12%

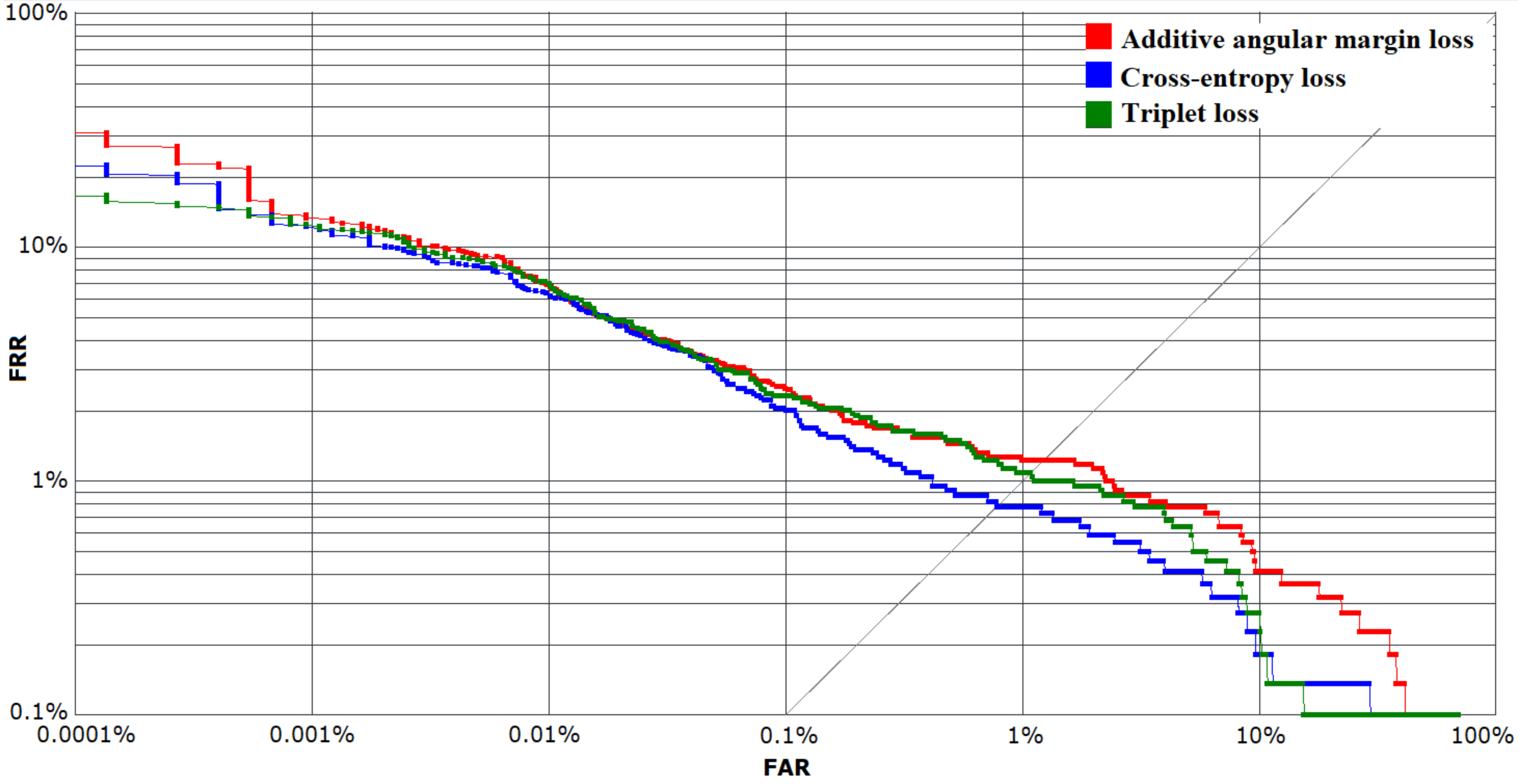

Figure 23. MEDS-II test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M small dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	3.355%	96.3%	51.9%	35.71%	23.83%
24995000	20000	24975000	2.57%	98.38%	53.9%	37.03%	22.57%
24995000	20000	24975000	3.027%	97.91%	50.85%	36.67%	23.99%

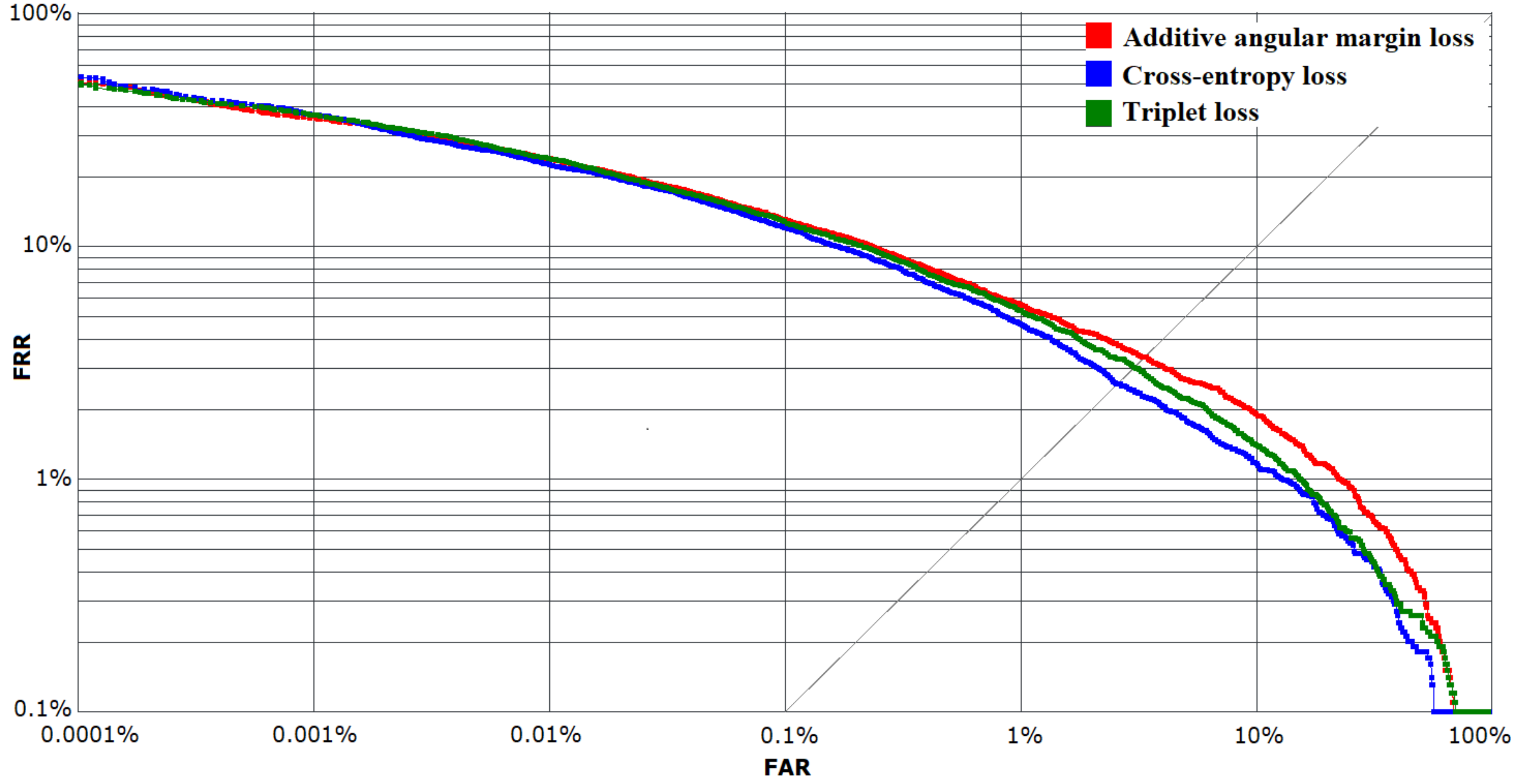

Figure 24. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M small dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.5%	1.633%	1.633%	1.633%	1.633%
6000	3000	3000	0.75%	6.8%	6.8%	6.8%	6.8%
6000	3000	3000	1.033%	8.767%	8.767%	8.767%	8.767%

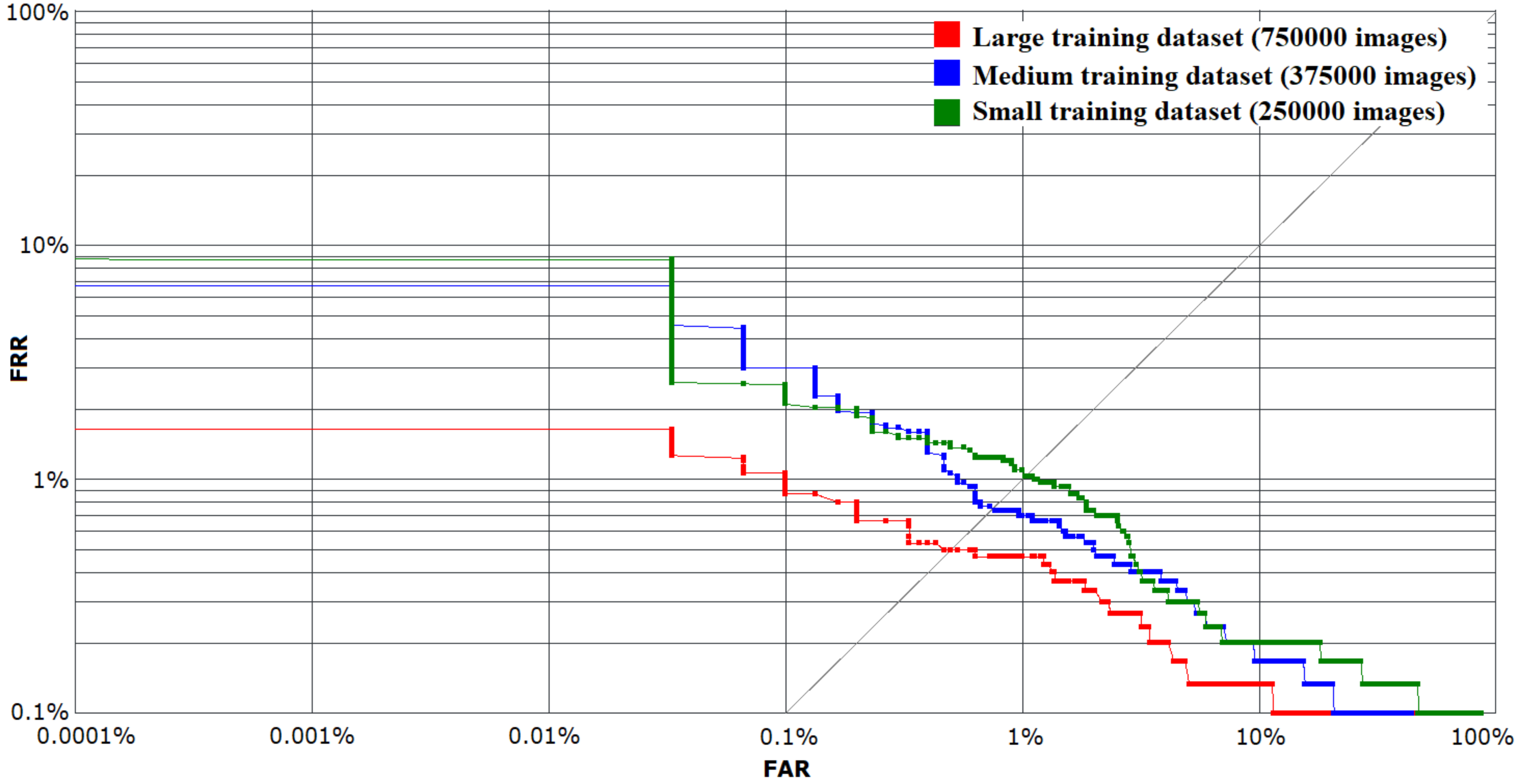

Figure 25. LFW test set DET curves of ResNet21 architecture trained with cross-entropy loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8126%	14.51%	14.51%	4.626%	1.769%
1477440	4410	1473030	0.8162%	21.32%	21.32%	13.74%	5.986%
1477440	4410	1473030	0.7948%	22.77%	22.77%	12.29%	6.349%

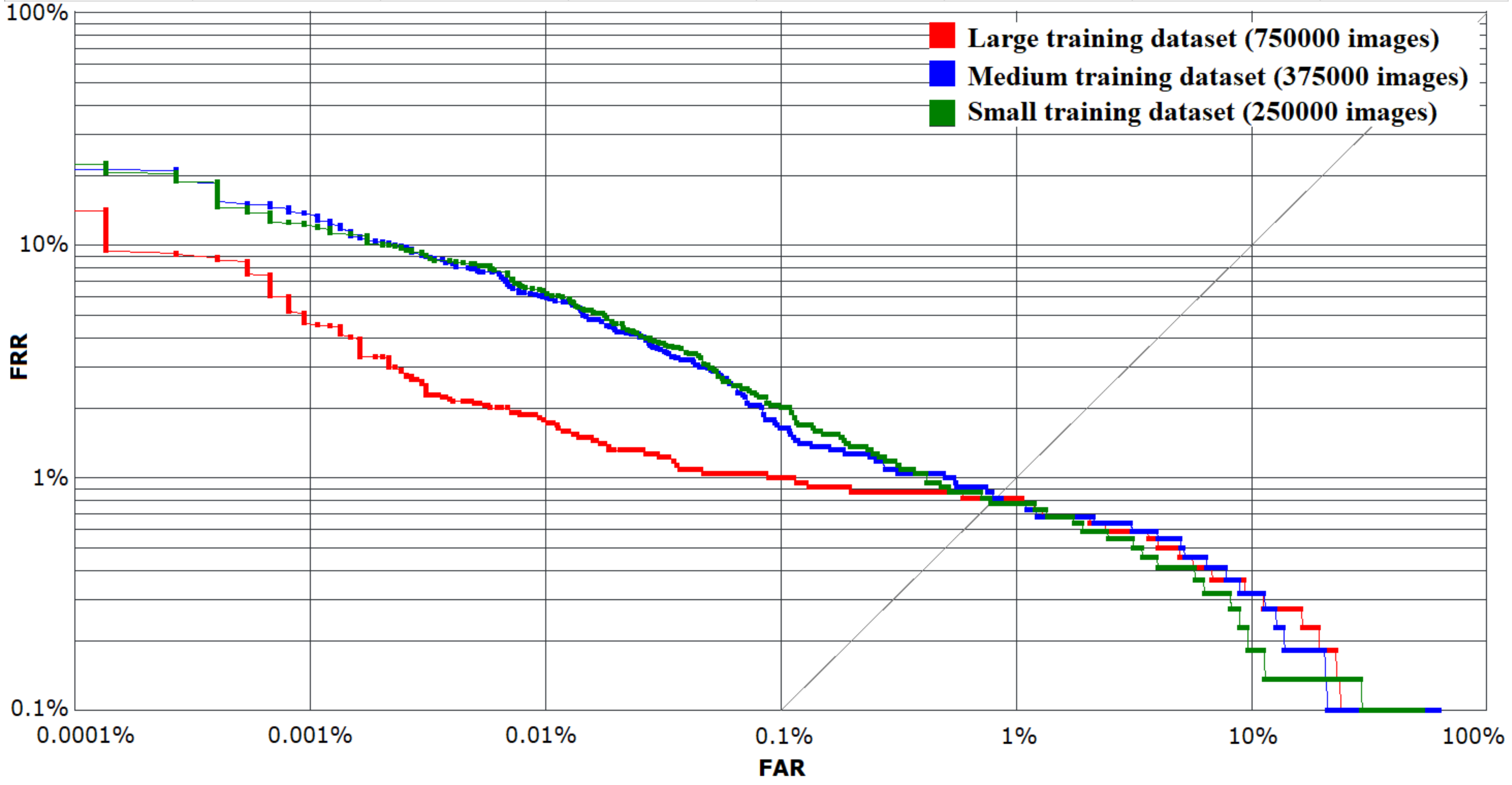

Figure 26. MEDS-II test set DET curves of ResNet21 architecture trained with cross-entropy loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	1.732%	97.3%	31.75%	18.75%	11.18%
24995000	20000	24975000	2.278%	98.33%	46.12%	31.66%	18.43%
24995000	20000	24975000	2.57%	98.38%	53.9%	37.03%	22.57%

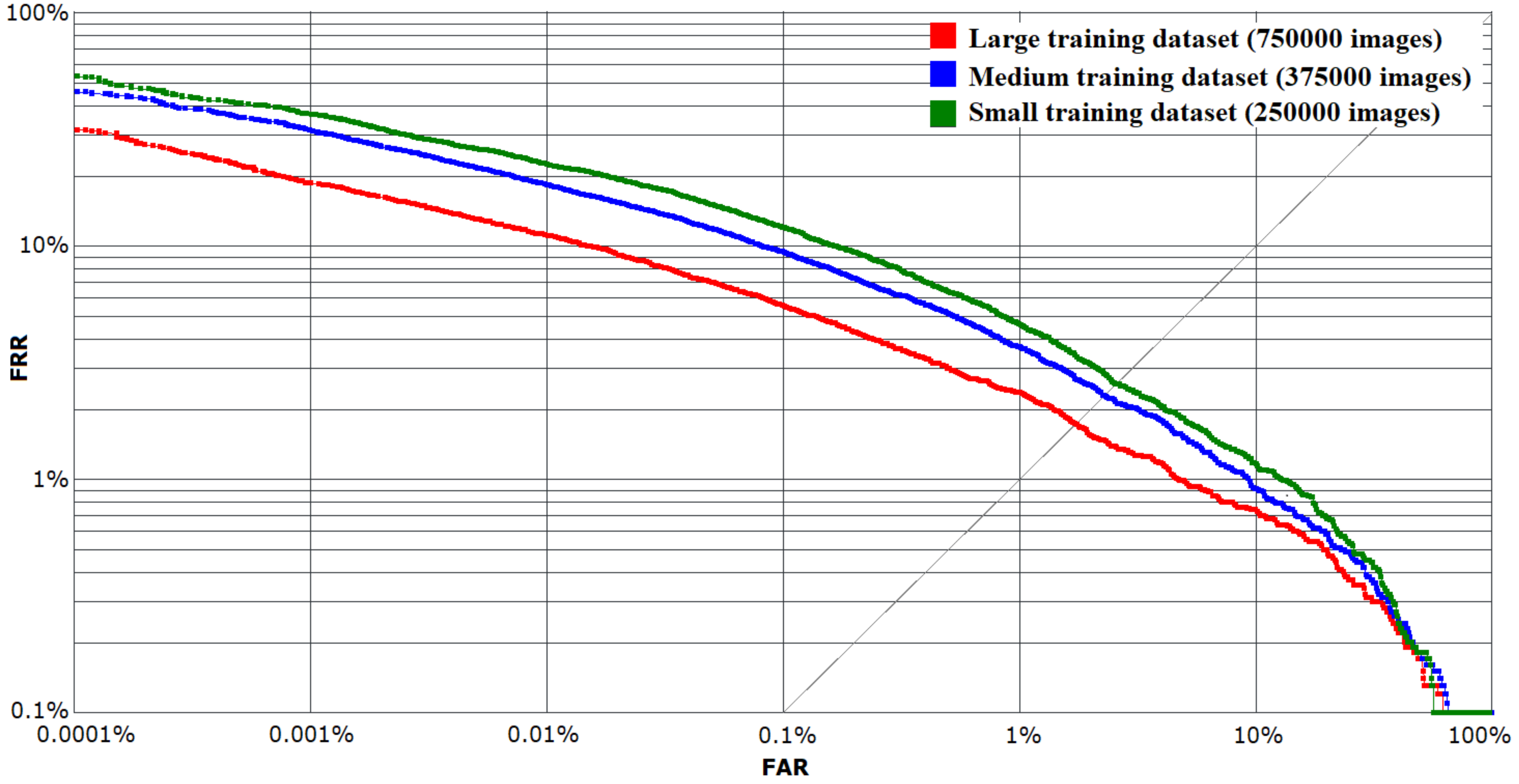

Figure 27. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with cross-entropy loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.6%	2.167%	2.167%	2.167%	2.167%
6000	3000	3000	0.8333%	12.57%	12.57%	12.57%	12.57%
6000	3000	3000	1.233%	8.2%	8.2%	8.2%	8.2%

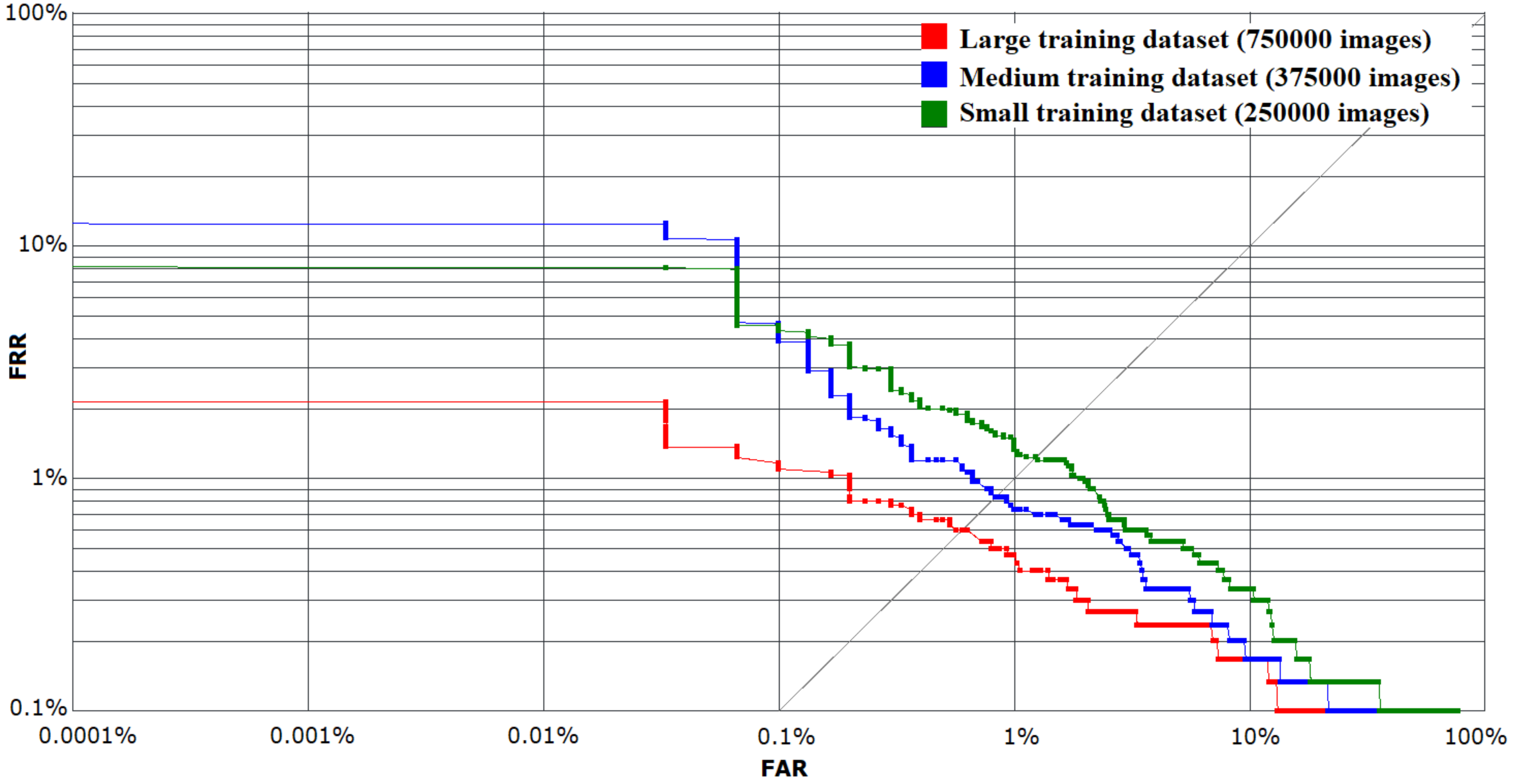

Figure 28. LFW test set DET curves of ResNet21 architecture trained with triplet loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8124%	15.83%	15.83%	5.397%	2.721%
1477440	4410	1473030	0.8526%	17.87%	17.87%	8.254%	4.58%
1477440	4410	1473030	1.086%	16.69%	16.69%	12.47%	7.12%

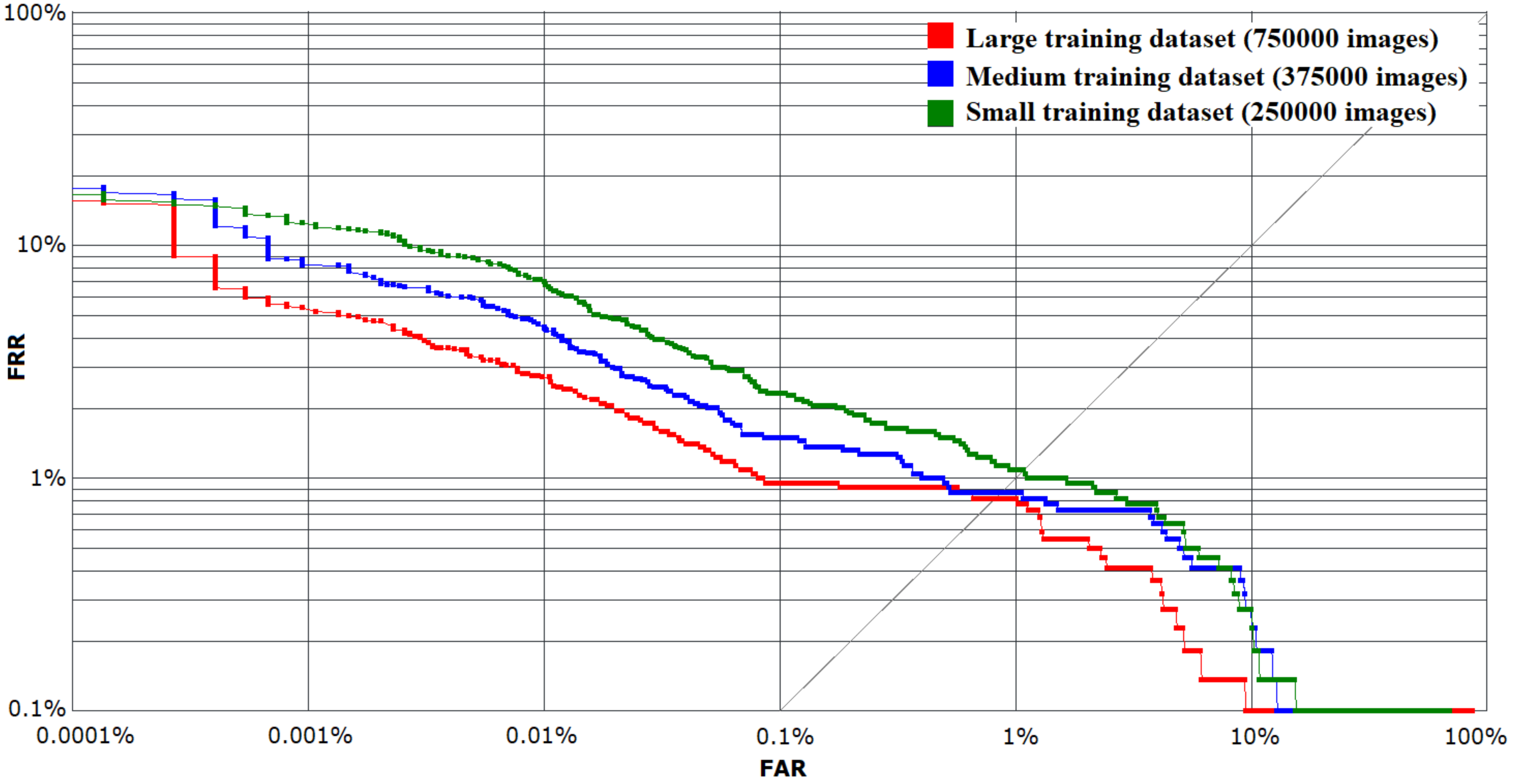

Figure 29. MEDS-II test set DET curves of ResNet21 architecture trained with triplet loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	1.831%	98.39%	36.62%	24.92%	13.76%
24995000	20000	24975000	2.327%	98.83%	40.48%	30.08%	18.56%
24995000	20000	24975000	3.027%	97.91%	50.85%	36.67%	23.99%

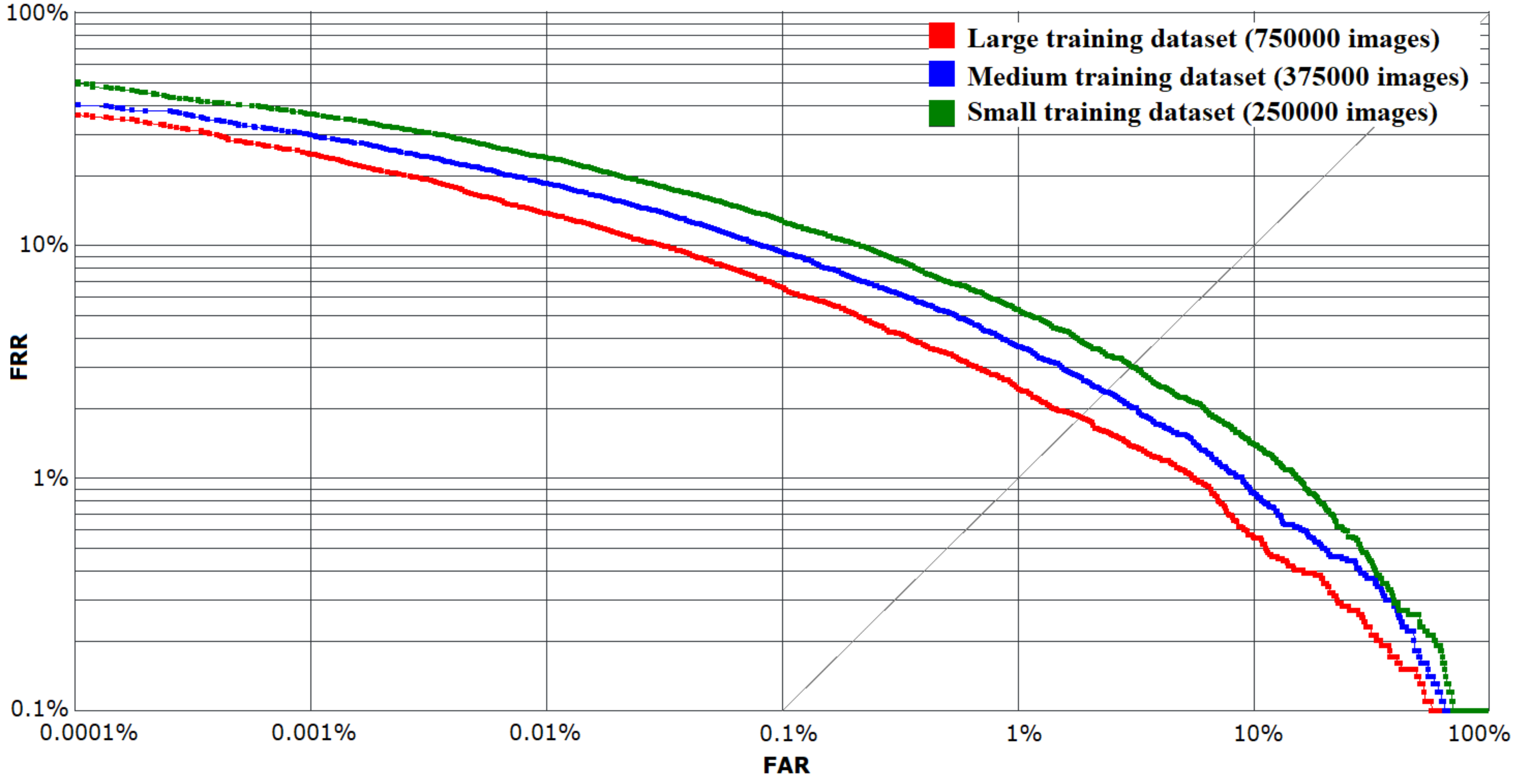

Figure 30. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with triplet loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.65%	3.067%	3.067%	3.067%	3.067%
6000	3000	3000	0.9667%	4.333%	4.333%	4.333%	4.333%
6000	3000	3000	1.25%	9.133%	9.133%	9.133%	9.133%

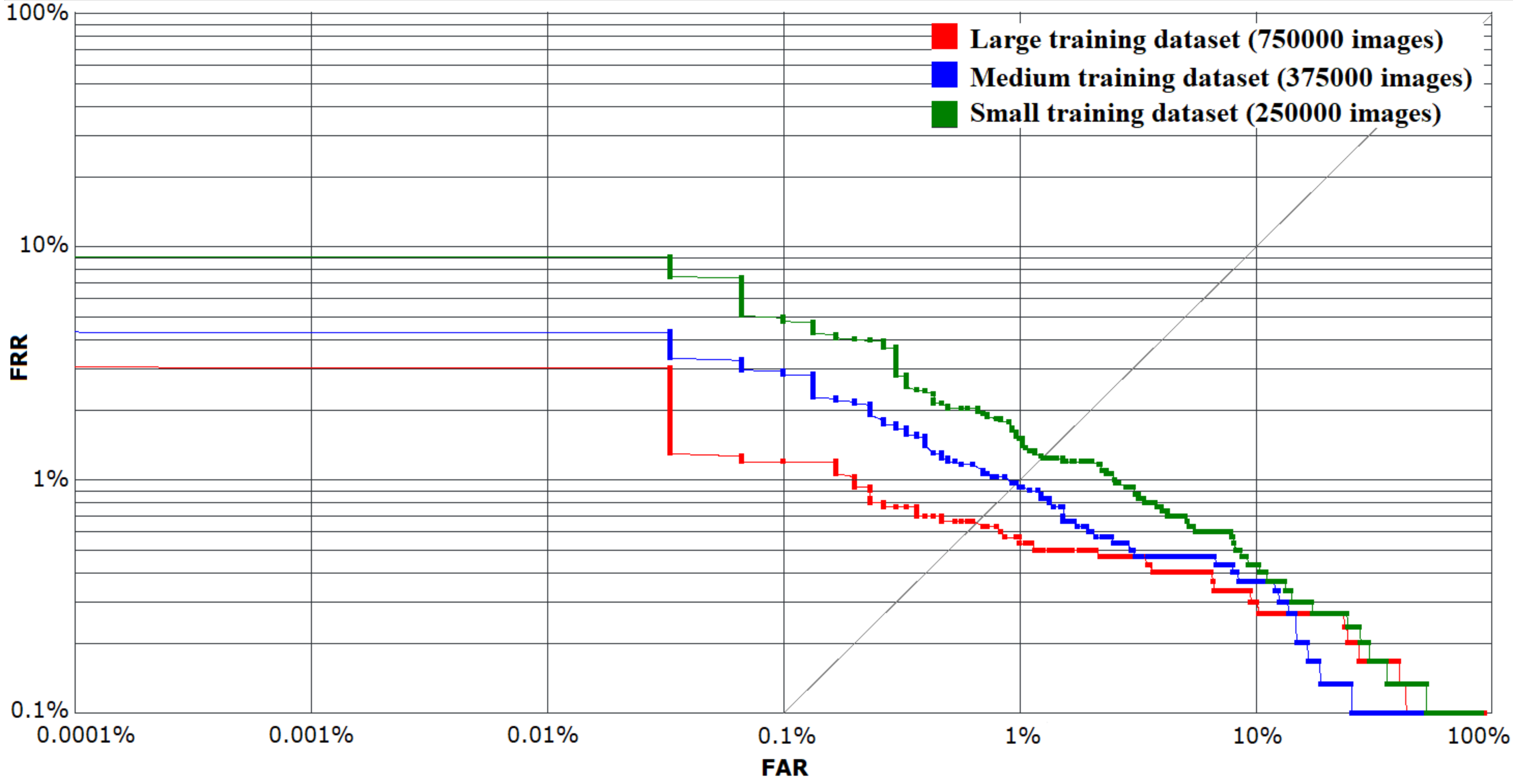

Figure 31. LFW test set DET curves of ResNet21 architecture trained with additive angular margin loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8084%	9.478%	9.478%	3.356%	1.769%
1477440	4410	1473030	0.9043%	11.25%	11.25%	7.347%	3.9%
1477440	4410	1473030	1.214%	31.38%	31.38%	13.38%	6.893%

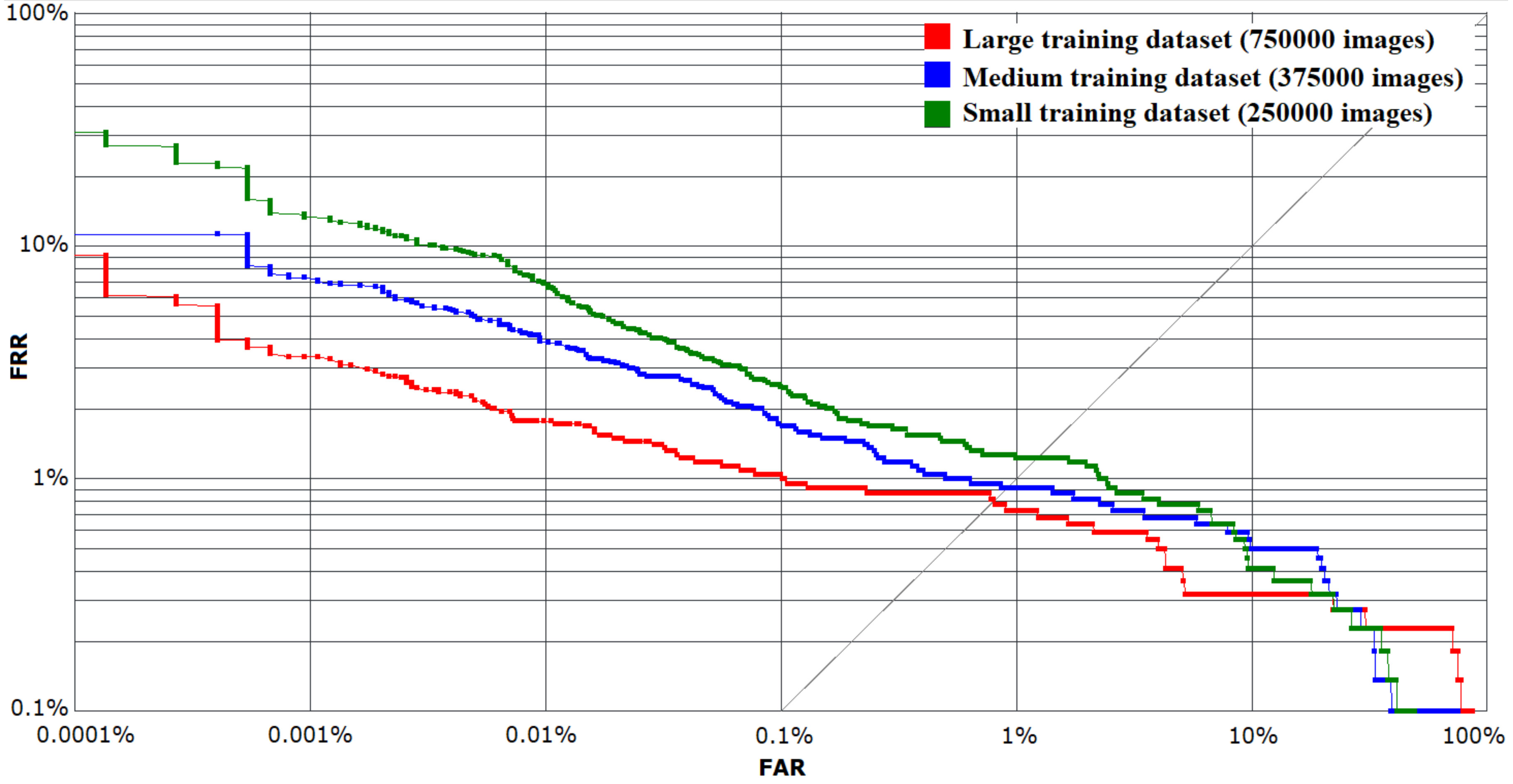

Figure 32. MEDS-II test set DET curves of ResNet21 architecture trained with additive angular margin loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	2.065%	96.58%	25.04%	16.58%	9.99%
24995000	20000	24975000	2.853%	98.56%	40.56%	27.69%	17.24%
24995000	20000	24975000	3.355%	96.3%	51.9%	35.71%	23.83%

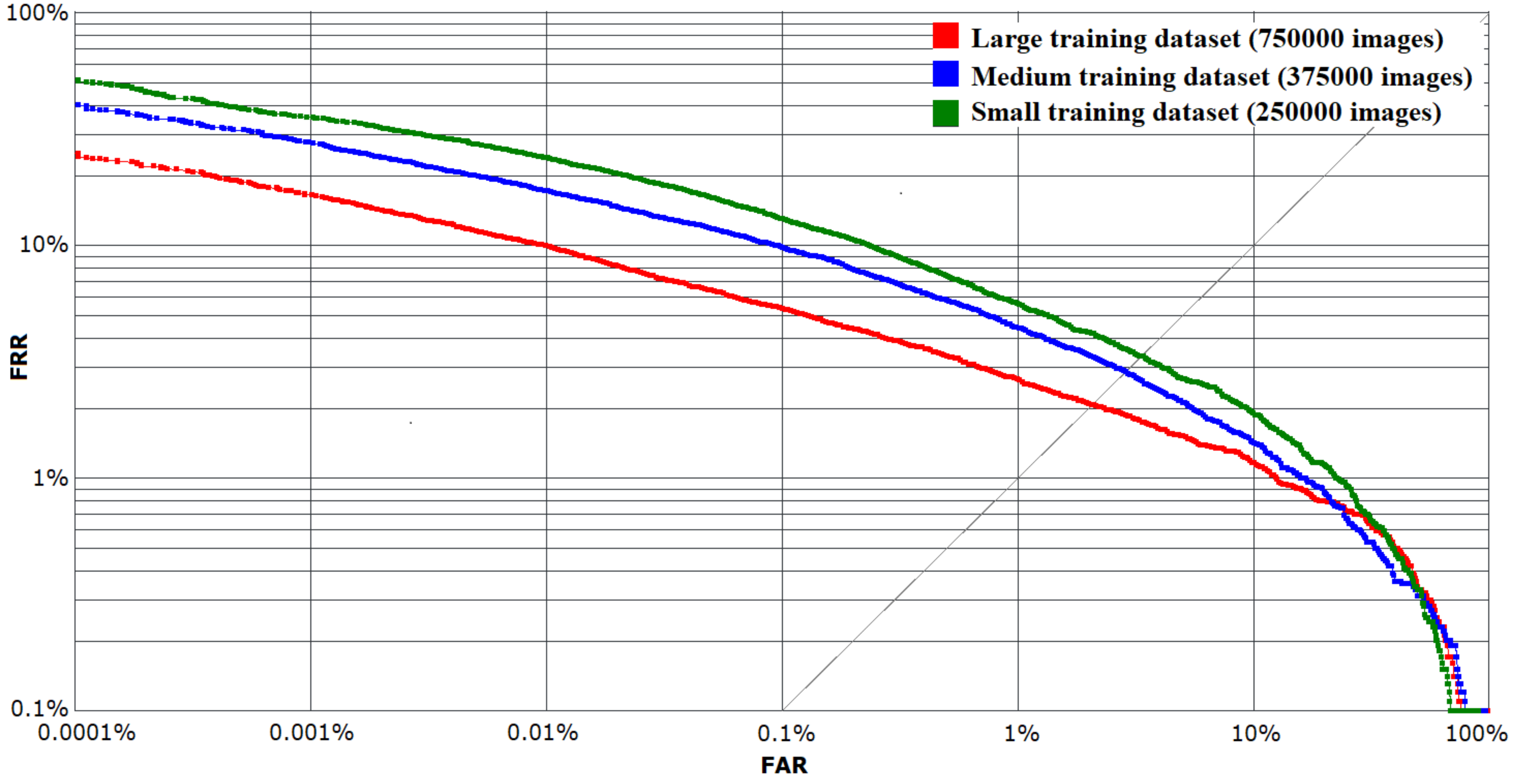

Figure 33. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin loss using MS-Celeb-1M large (red), medium (blue), small (green) datasets.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.5%	1.633%	1.633%	1.633%	1.633%
6000	3000	3000	0.45%	2.033%	2.033%	2.033%	2.033%

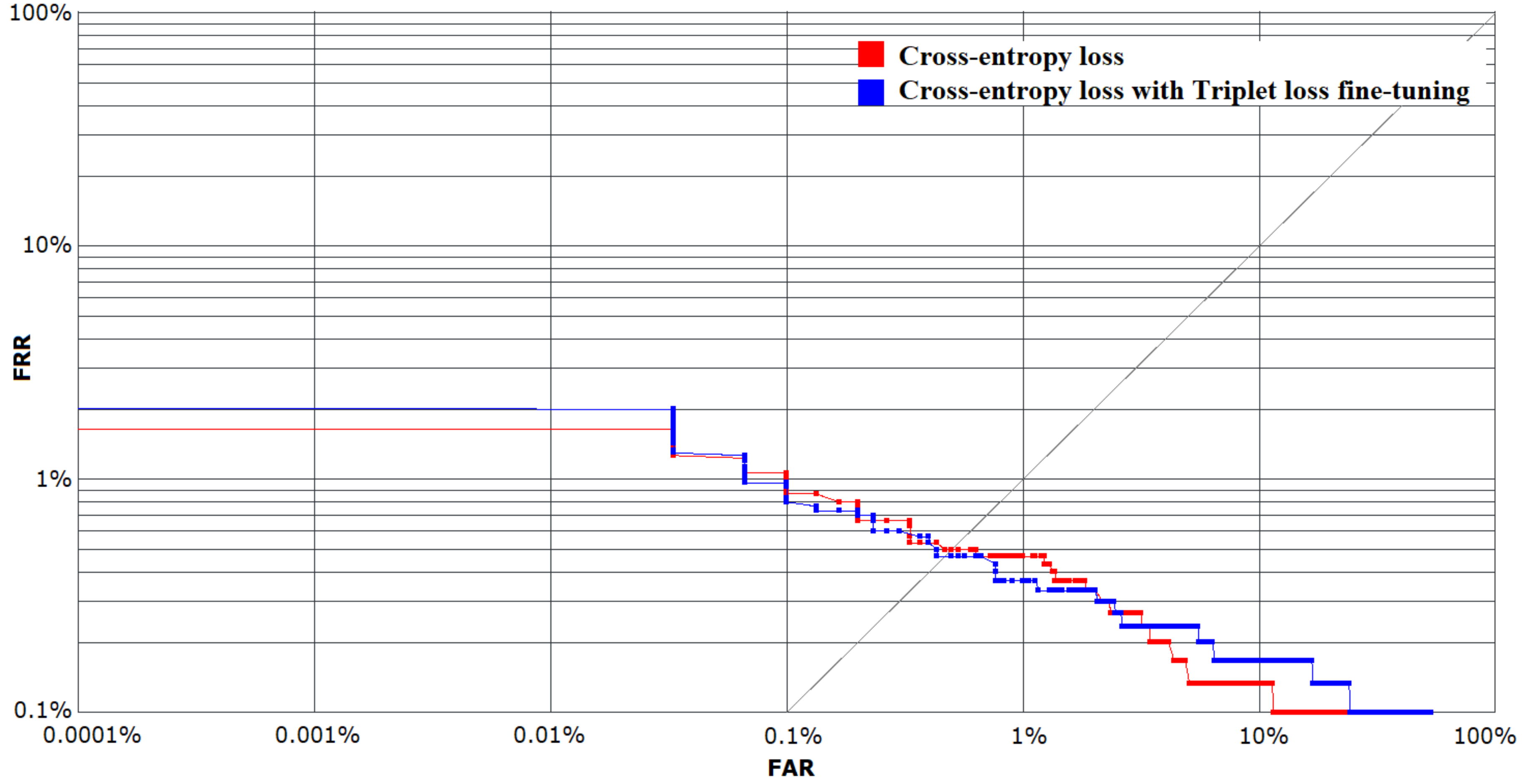

Figure 34. LFW test set DET curves of ResNet21 architecture trained with cross-entropy loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8126%	14.51%	14.51%	4.626%	1.769%
1477440	4410	1473030	0.7243%	10.29%	10.29%	4.263%	1.497%

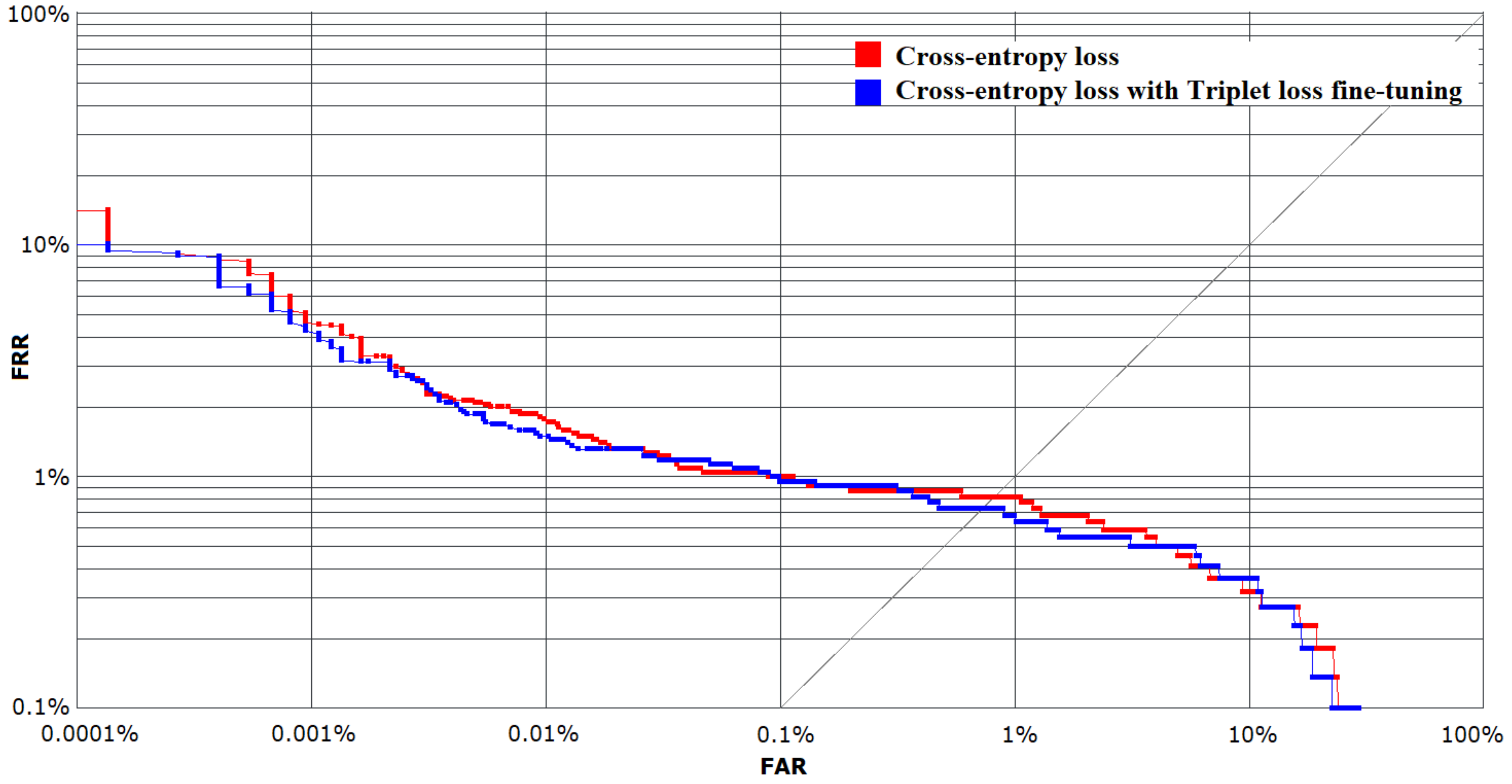

Figure 35. MEDS-II test set DET curves of ResNet21 architecture trained with cross-entropy loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	1.732%	97.3%	31.75%	18.75%	11.18%
24995000	20000	24975000	1.614%	97.24%	29.25%	17.68%	9.91%

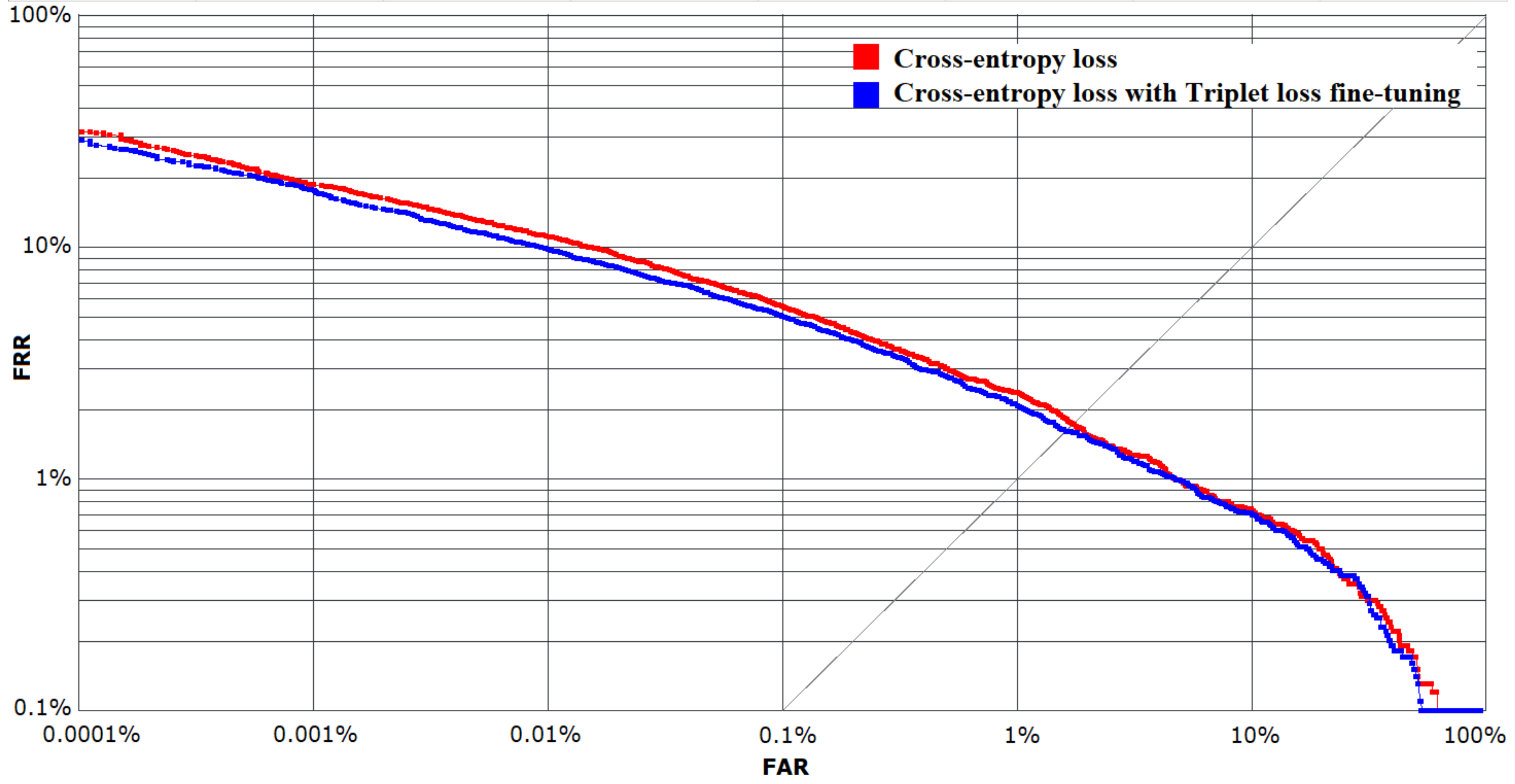

Figure 36. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with cross-entropy loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.65%	3.067%	3.067%	3.067%	3.067%
6000	3000	3000	0.6667%	2.733%	2.733%	2.733%	2.733%

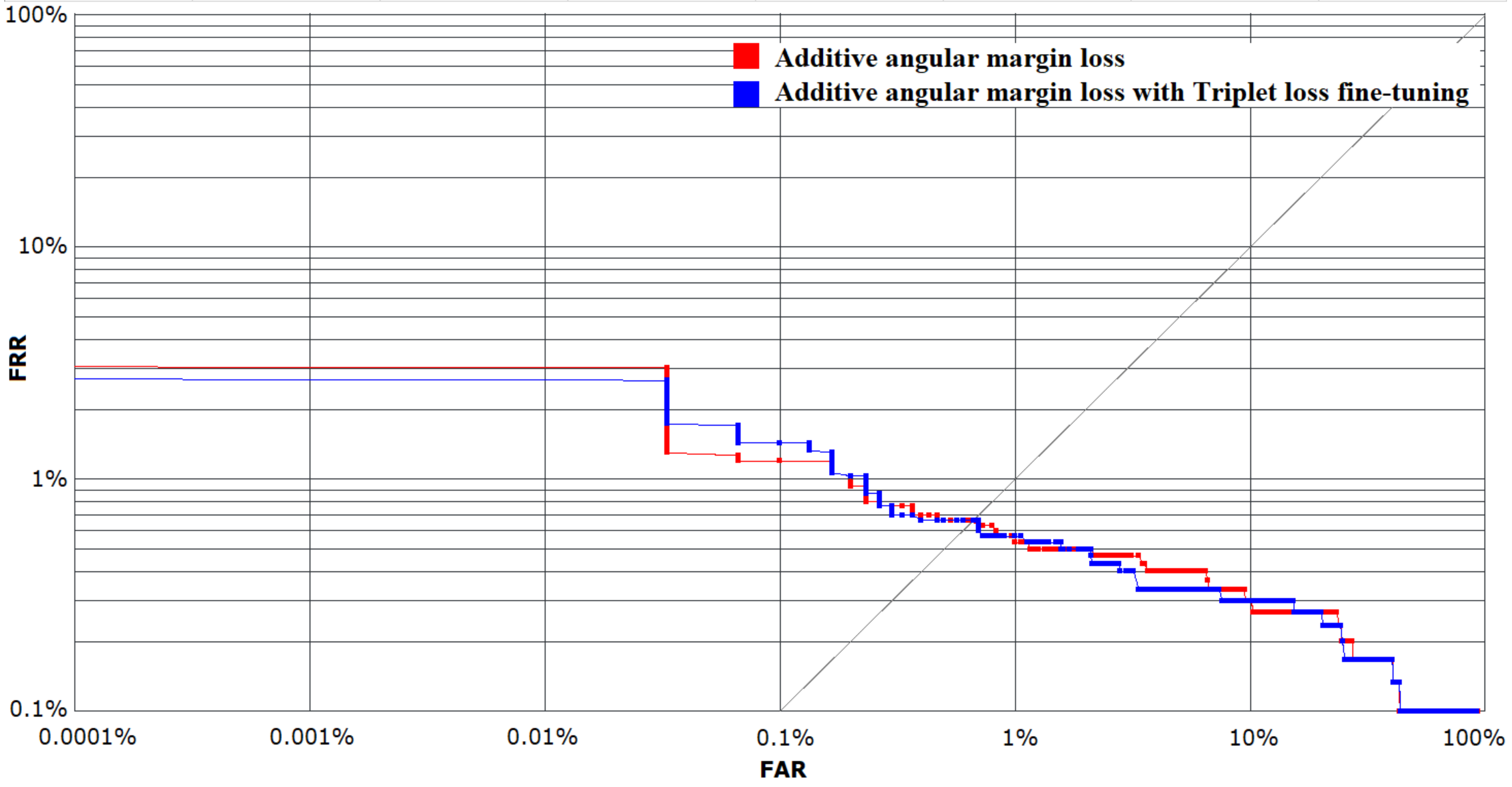

Figure 37. LFW test set DET curves of ResNet21 architecture trained with additive angular margin loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8084%	9.478%	9.478%	3.356%	1.769%
1477440	4410	1473030	0.8599%	9.161%	9.161%	3.81%	2.041%

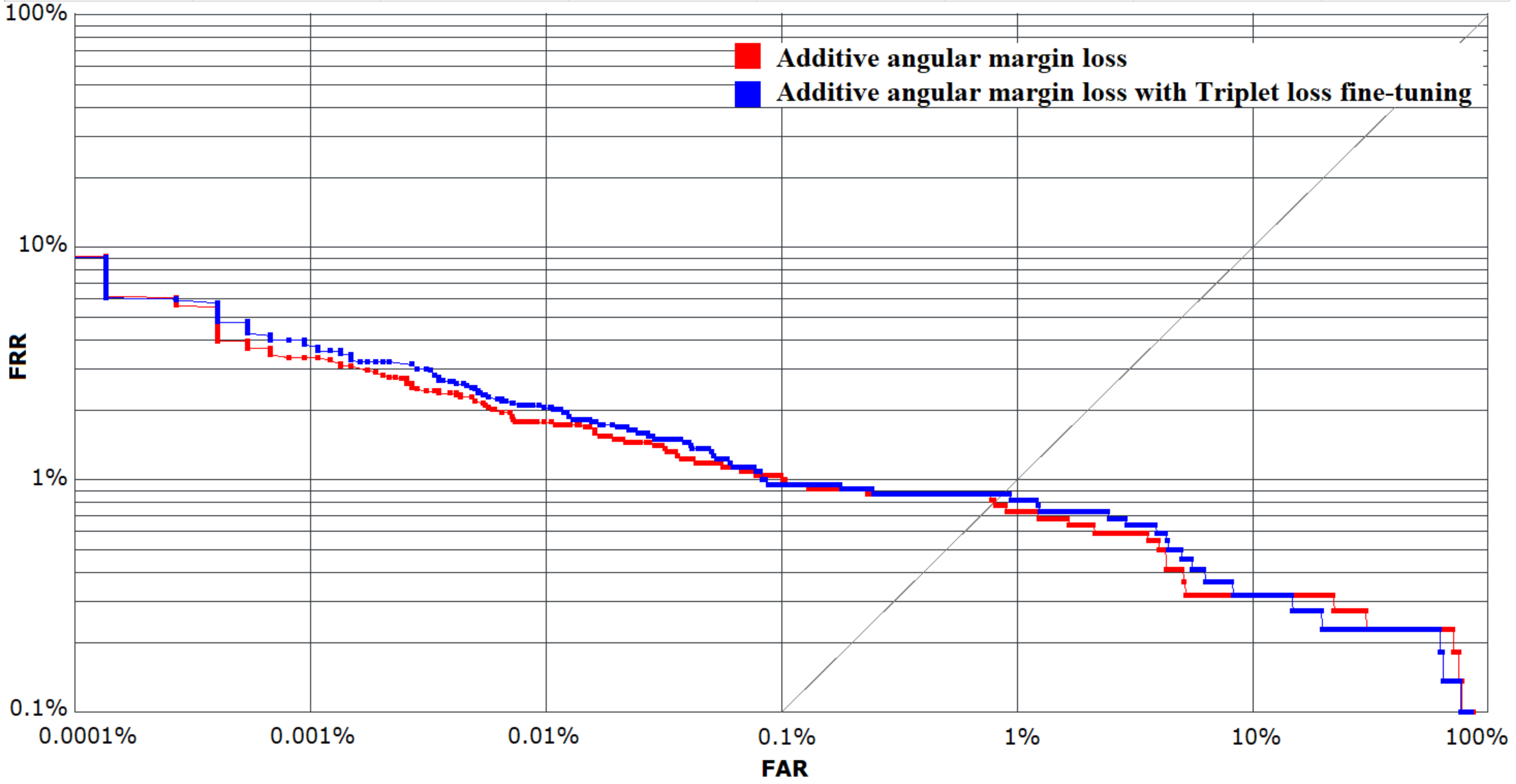

Figure 38. MEDS-II test set DET curves of ResNet21 architecture trained with additive angular margin loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	2.065%	96.58%	25.04%	16.58%	9.99%
24995000	20000	24975000	2.032%	95.61%	25.31%	17.14%	10.17%

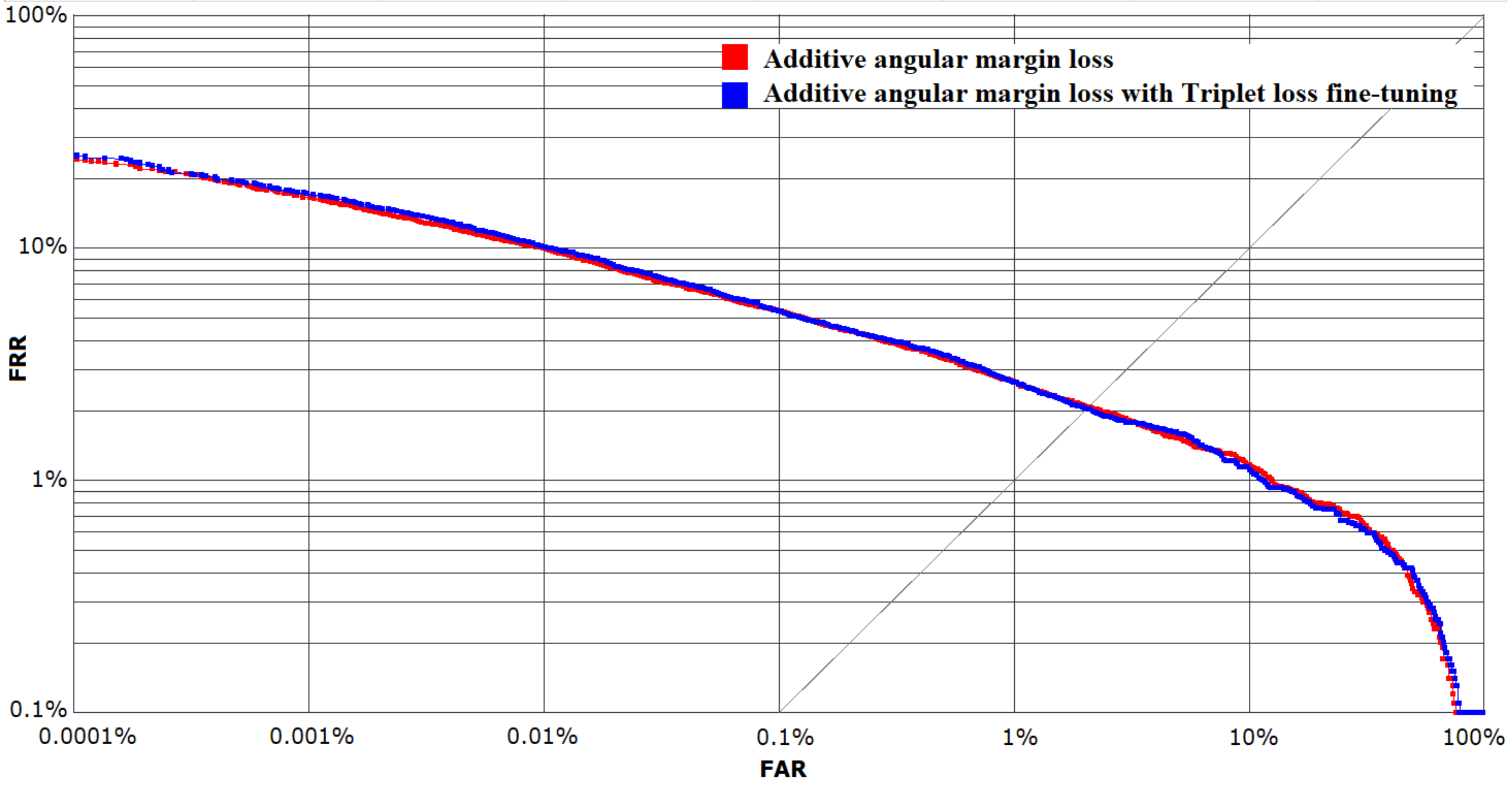

Figure 39. MS-Celeb-1M test set DET curves of ResNet21 architecture trained with additive angular margin loss (red) and additionally fine-tuned with triplet loss (blue) using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.6167%	2.8%	2.8%	2.8%	2.8%
6000	3000	3000	0.5333%	3.3%	3.3%	3.3%	3.3%
6000	3000	3000	0.6333%	2.767%	2.767%	2.767%	2.767%

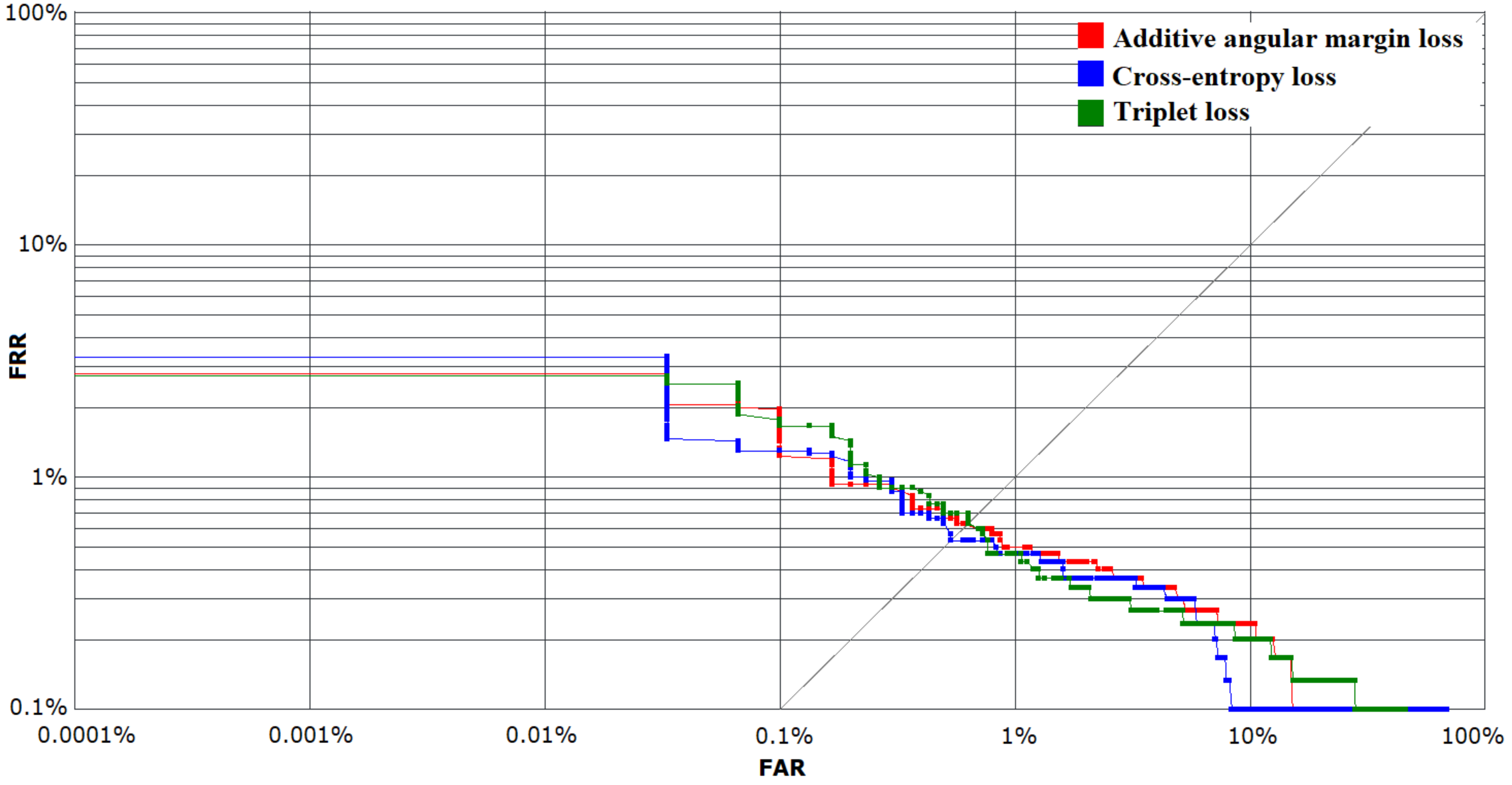

Figure 40. LFW test set DET curves of ResNet13 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.6734%	3.673%	3.673%	2.721%	1.224%
1477440	4410	1473030	0.8385%	5.896%	5.896%	3.311%	1.723%
1477440	4410	1473030	0.7178%	9.705%	9.705%	6.304%	3.129%

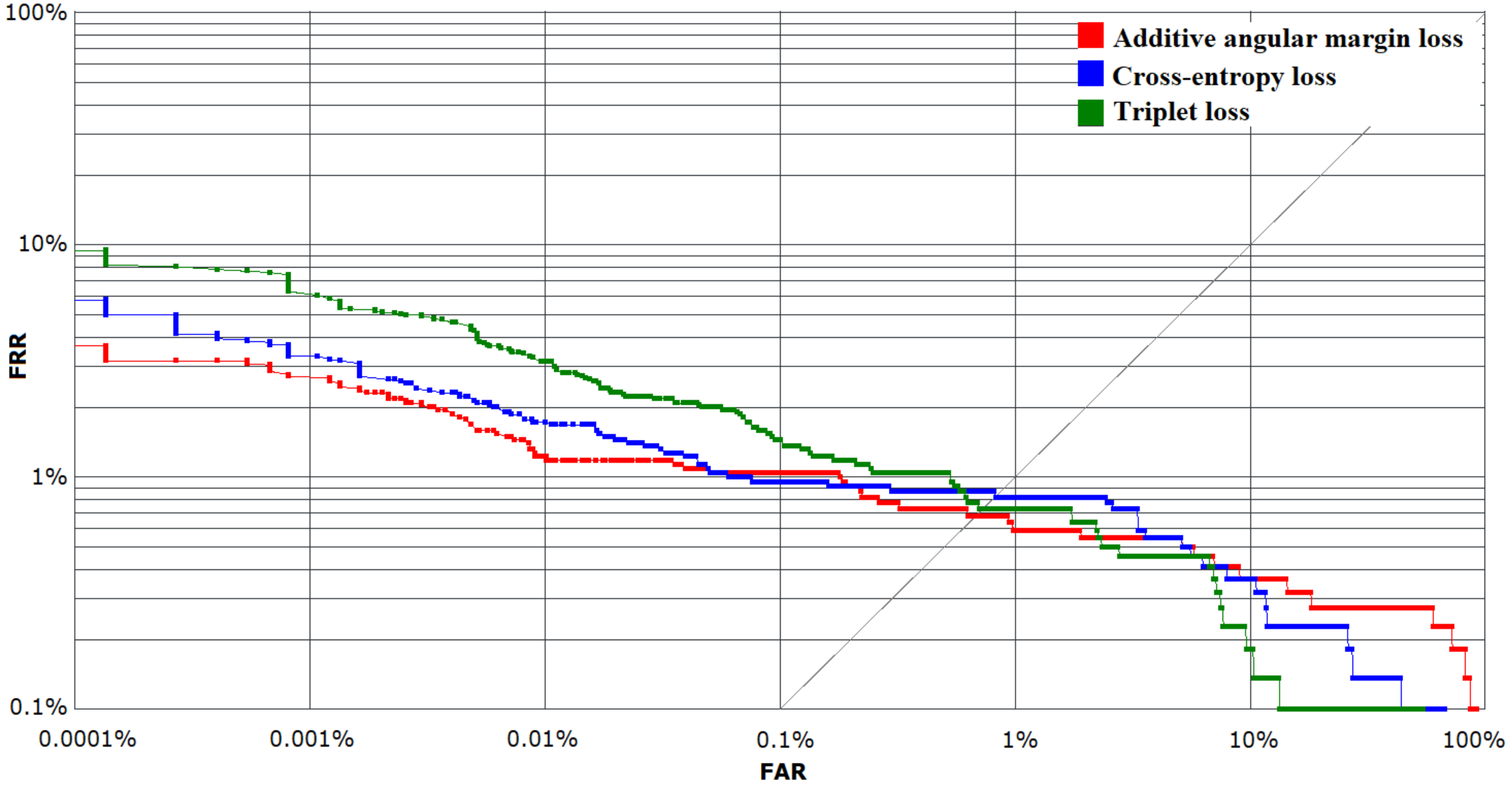

Figure 41. MEDS-II test set DET curves of ResNet13 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	2.24%	97.12%	26.86%	16.4%	10.19%
24995000	20000	24975000	2.019%	98.7%	36.66%	20.61%	12.31%
24995000	20000	24975000	1.906%	98.47%	42.22%	25.9%	15.36%

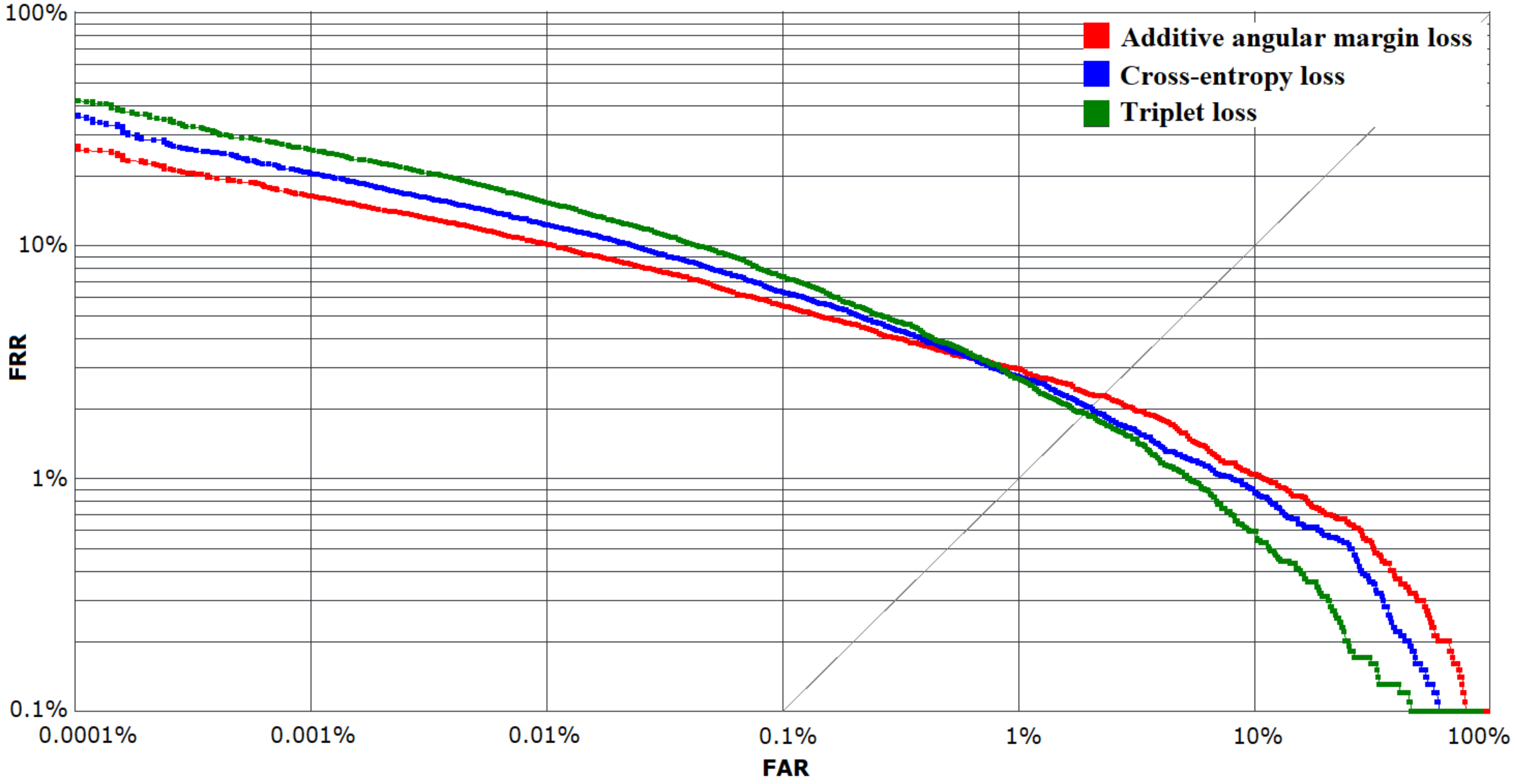

Figure 42. MS-Celeb-1M test set DET curves of ResNet13 architecture trained with additive angular margin (red), cross-entropy (blue), triplet (green) losses using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.5%	1.663%	1.663%	1.663%	1.663%
1477440	4410	1473030	0.5333%	3.3%	3.3%	3.3%	3.3%

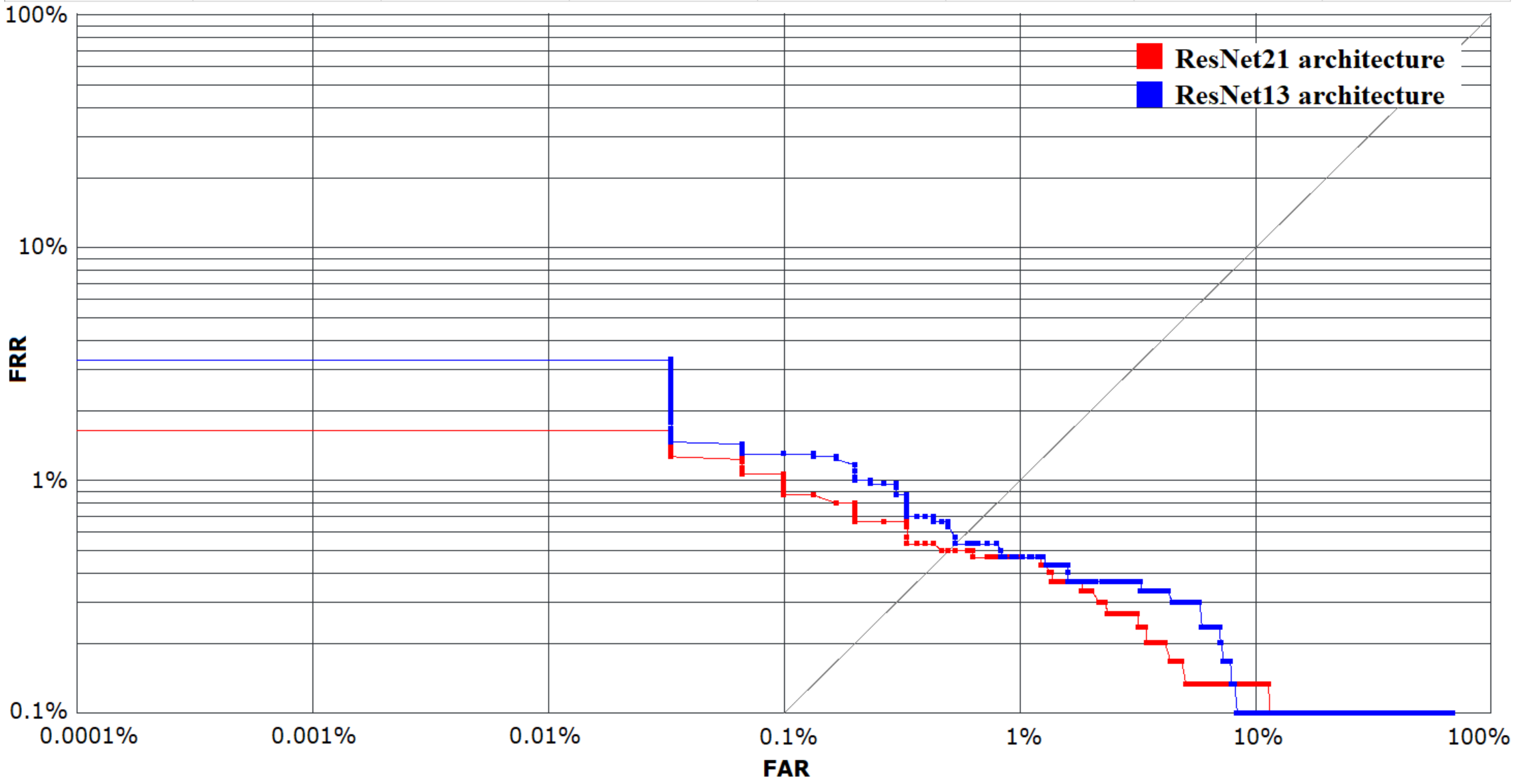

Figure 43. LFW test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with cross-entropy loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8126%	14.51%	14.51%	4.626%	1.769%
1477440	4410	1473030	0.8385%	5.896%	5.896%	3.311%	1.723%

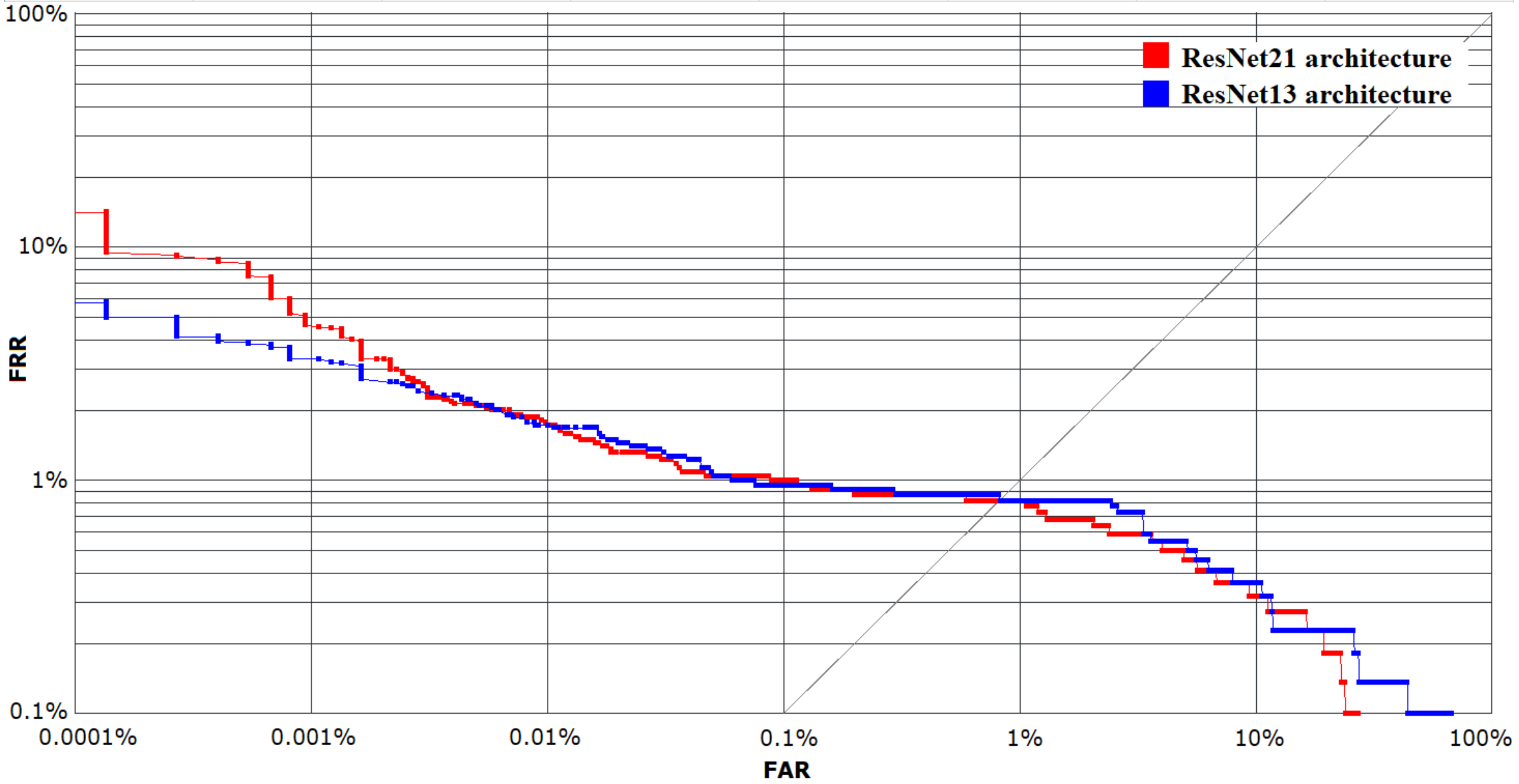

Figure 44. MEDS-II test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with cross-entropy loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	1.732%	97.3%	31.75%	18.75%	11.18%
24995000	20000	24975000	2.019%	98.7%	36.66%	20.61%	12.31%

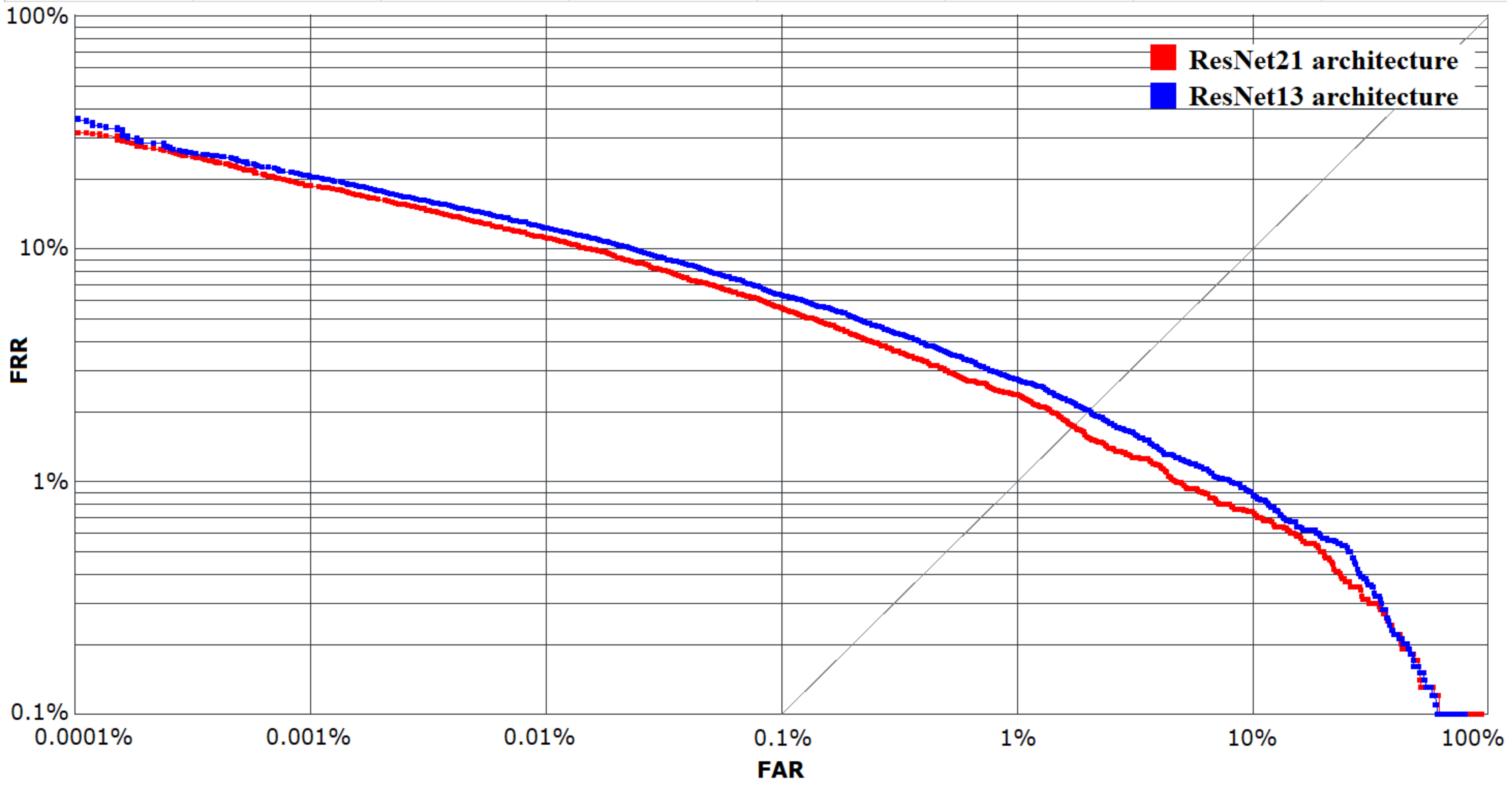

Figure 45. MS-Celeb-1M test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with cross-entropy loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.6%	2.167%	2.167%	2.167%	2.167%
6000	3000	3000	0.6333%	2.767%	2.767%	2.767%	2.767%

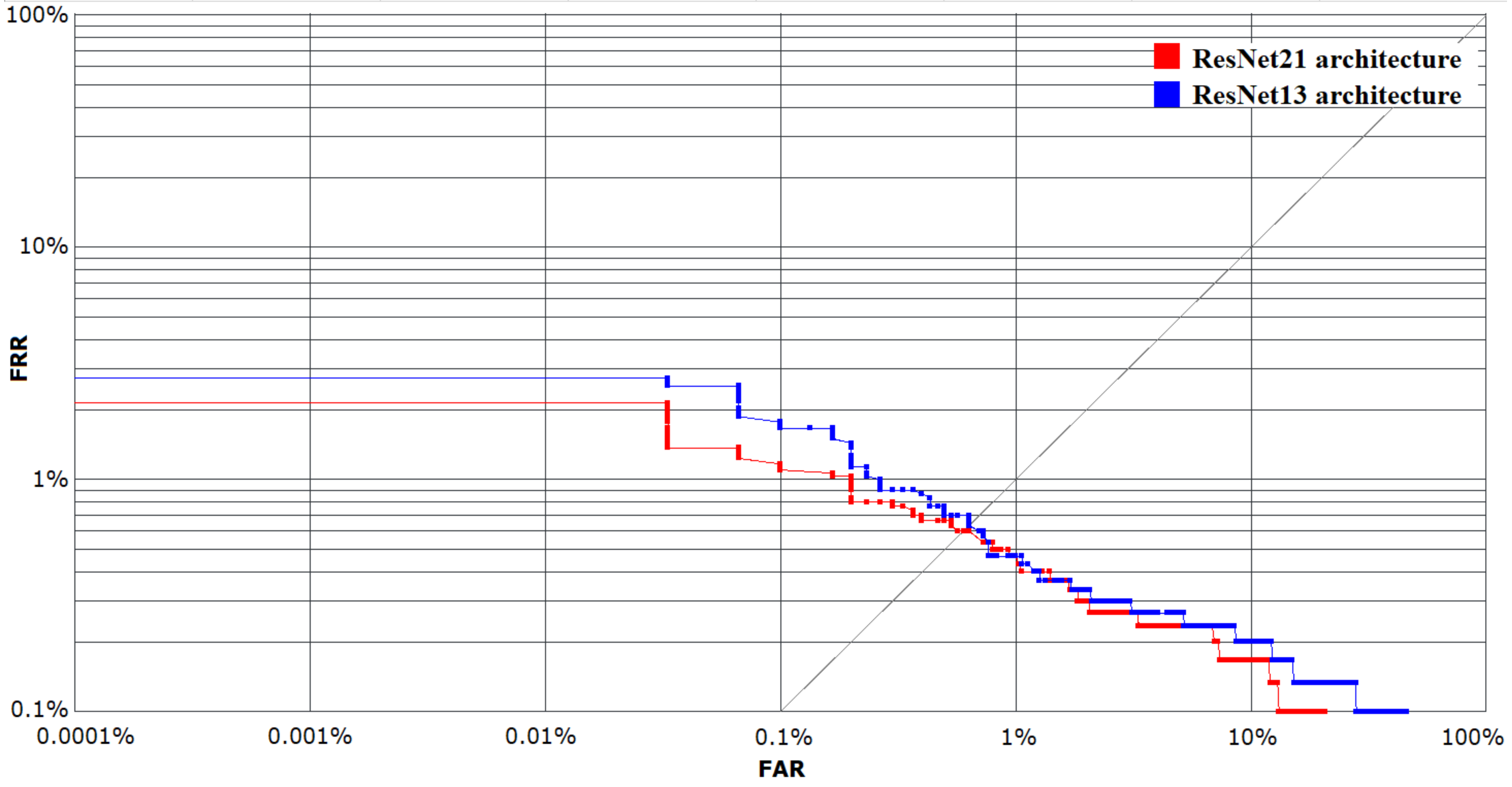

Figure 46. LFW test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with triplet loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8124%	15.83%	15.83%	5.397%	2.712%
1477440	4410	1473030	0.7178%	9.705%	9.705%	6.304%	3.129%

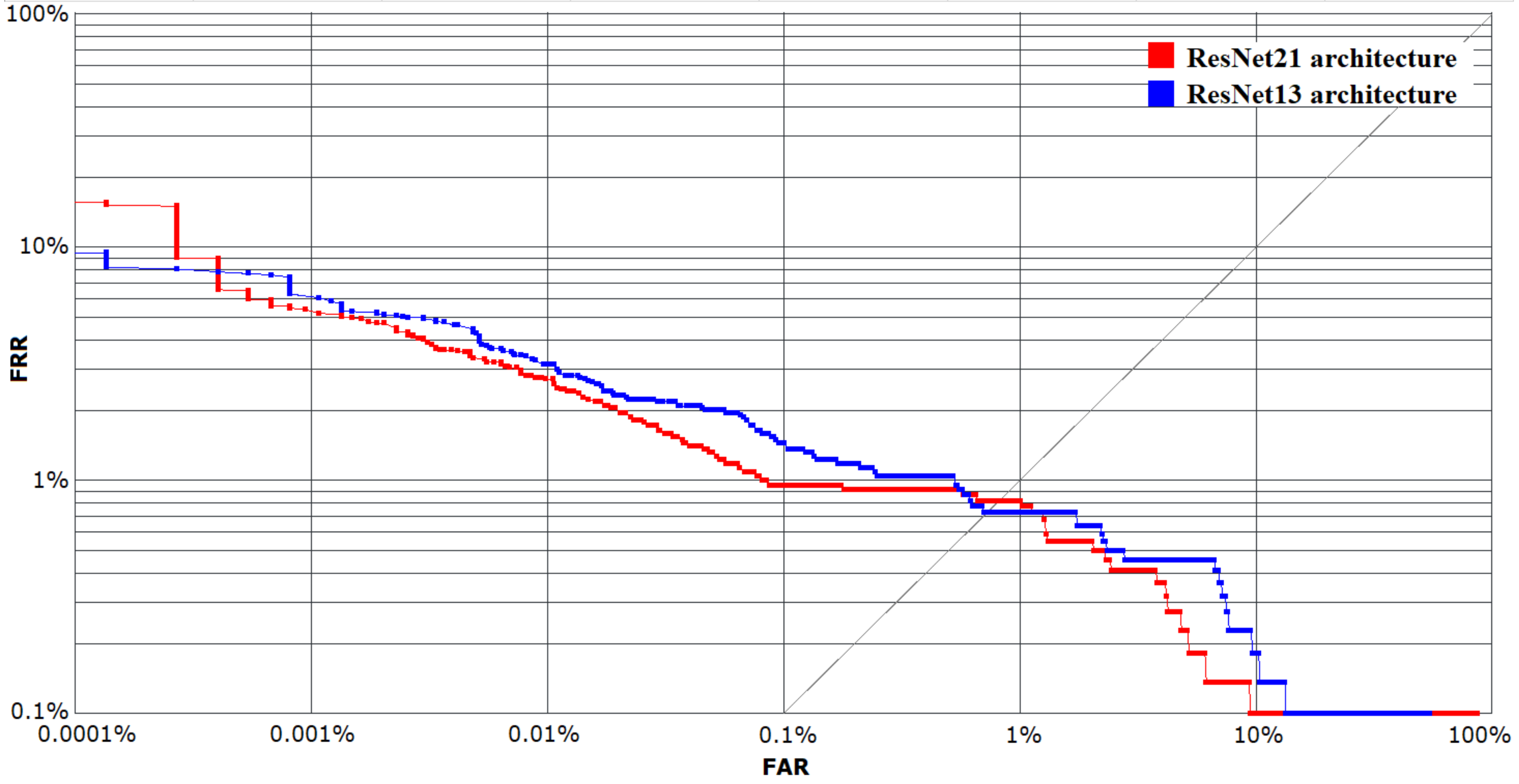

Figure 47. MEDS-II test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with triplet loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	1.831%	98.39%	36.62%	24.92%	13.76%
24995000	20000	24975000	1.906%	98.47%	42.22%	25.9%	15.36%

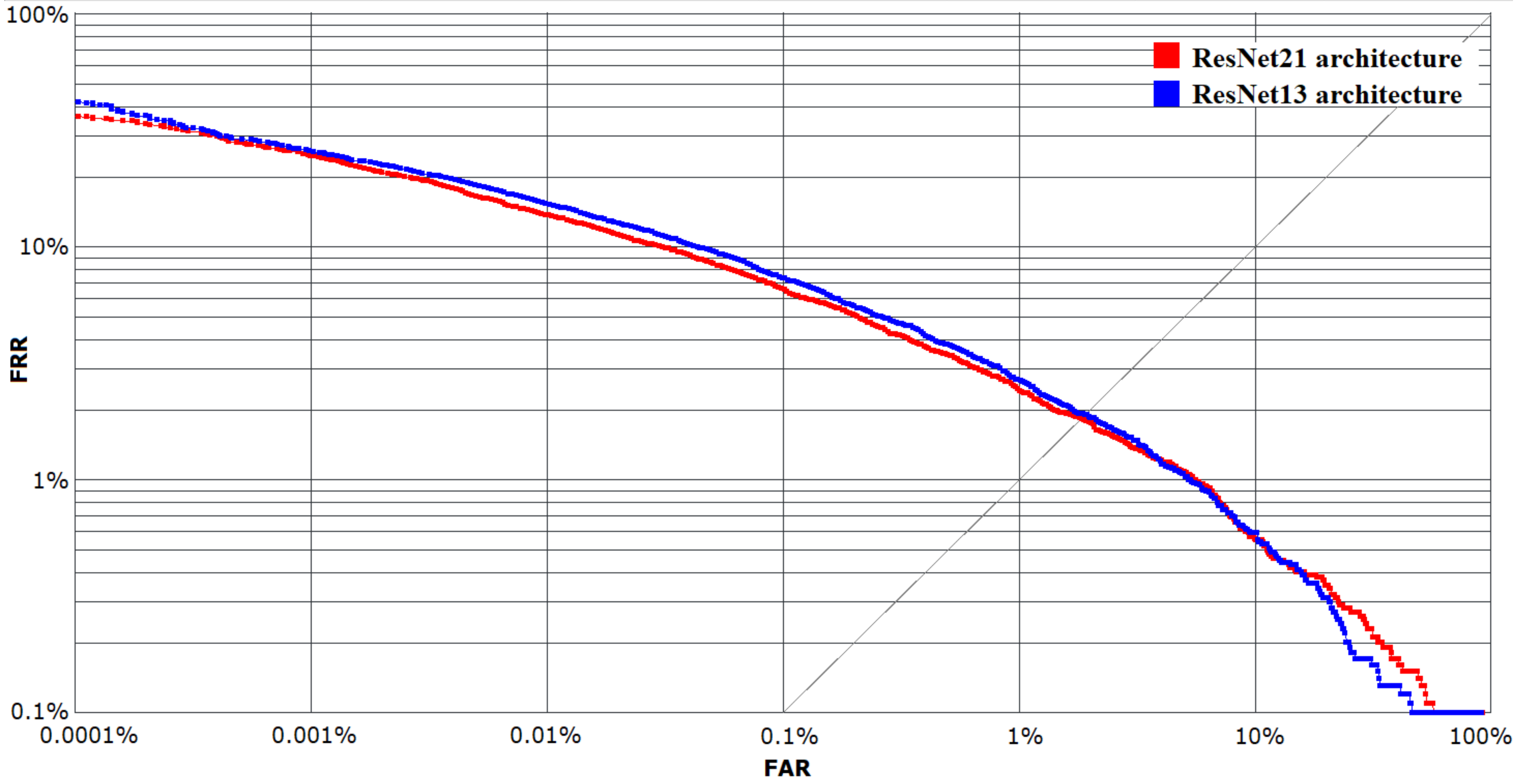

Figure 48. MS-Celeb-1M test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with triplet loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
6000	3000	3000	0.65%	3.067%	3.067%	3.067%	3.067%
6000	3000	3000	0.6167%	2.8%	2.8%	2.8%	2.8%

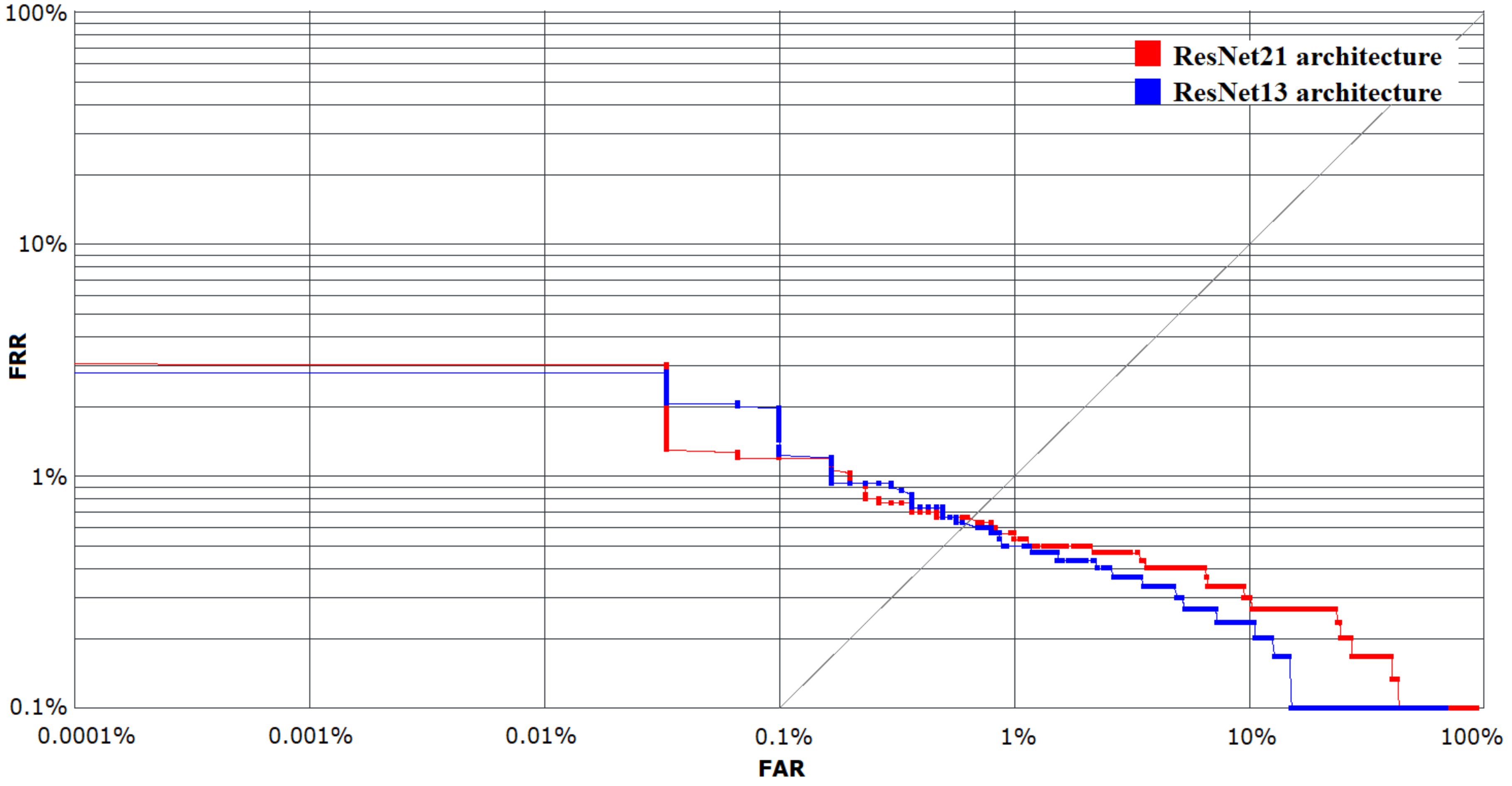

Figure 49. LFW test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with additive angular margin loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
1477440	4410	1473030	0.8084%	9.478%	9.478%	3.356%	1.769%
1477440	4410	1473030	0.6734%	3.673%	3.673%	2.721%	1.224%

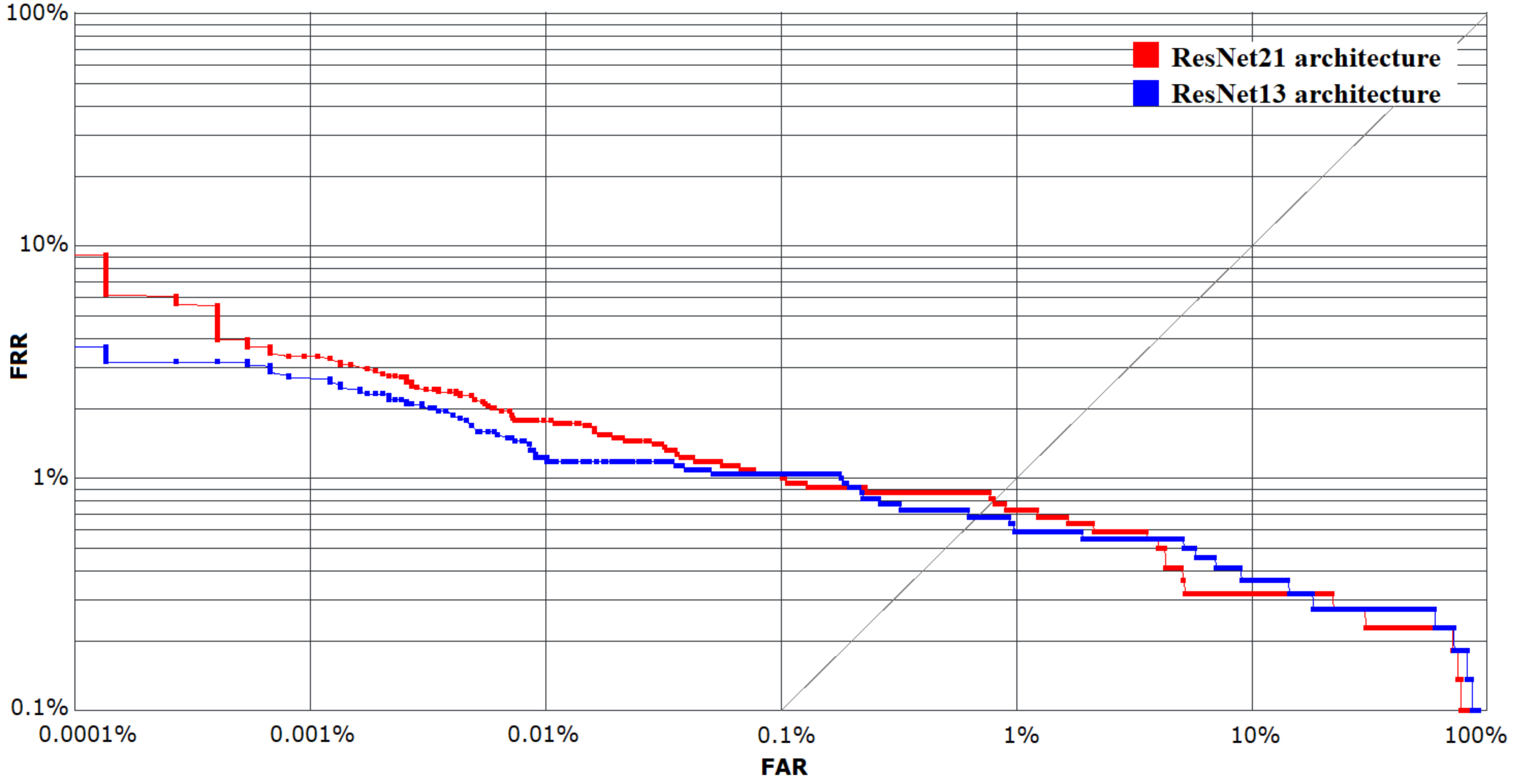

Figure 50. MEDS-II test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with additive angular margin loss using MS-Celeb-1M large dataset.

Pairs	Genuine pairs	Impostor pairs	EER	FRR@FAR=0%	FRR@FAR=0.0001%	FRR@FAR=0.001%	FRR@FAR=0.01%
24995000	20000	24975000	2.065%	96.58%	25.04%	16.58%	9.99%
24995000	20000	24975000	2.24%	97.12%	26.86%	16.4%	10.19%

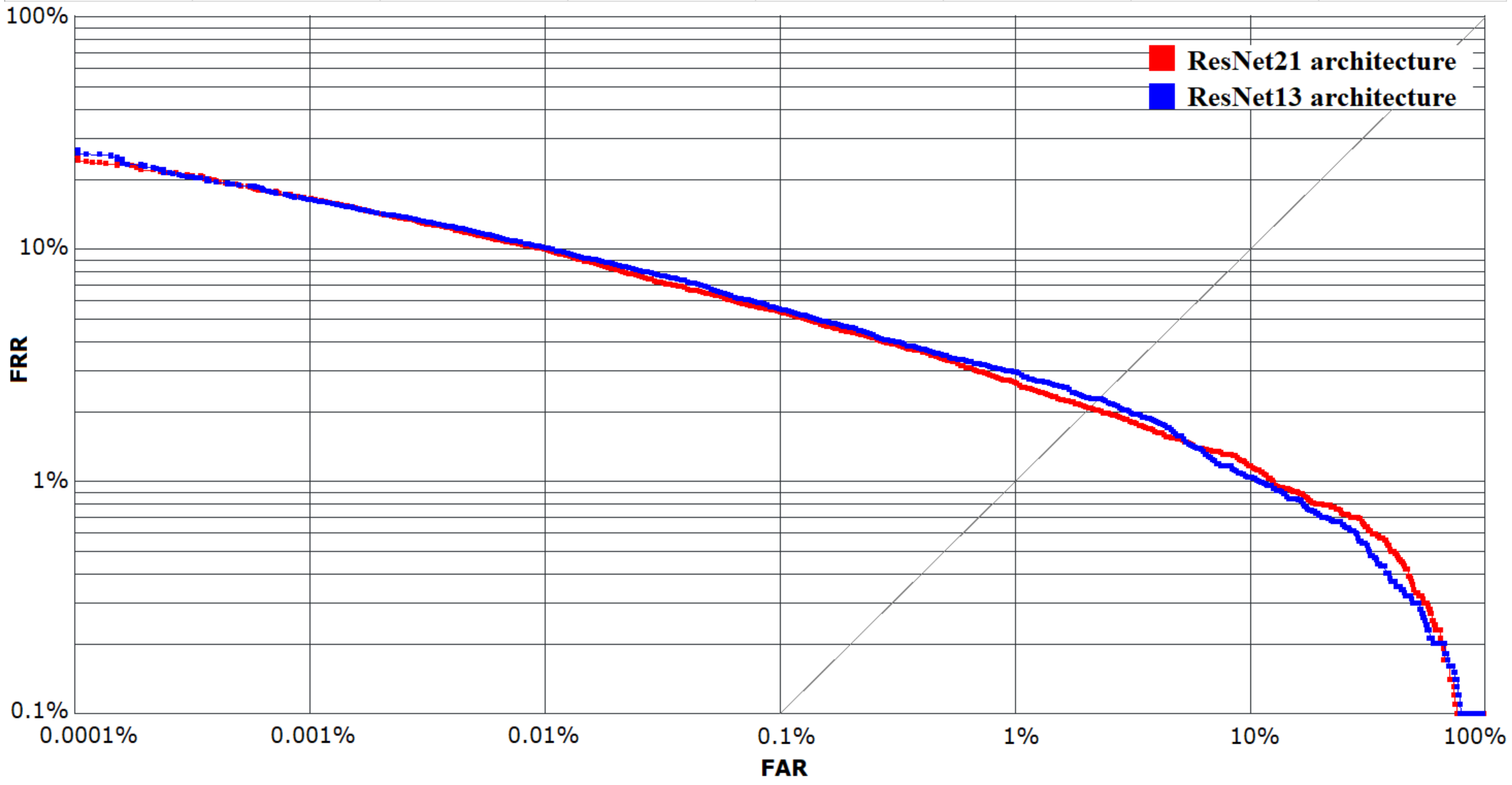

Figure 51. MS-Celeb-1M test set DET curves of ResNet21 (red), ResNet13 (blue) architectures trained with additive angular margin loss using MS-Celeb-1M large dataset.