Impact of Sampling Techniques on Siamese Neural Networks

Kartikay Srivastava
1599127

Lara Breuer
1221417

Abstract

Over the past few years Siamese Neural Networks have shown promising results in the field of image recognition problems where there is scarcity of training data. In this paper we investigate the impact of different sampling techniques on the performance of Siamese Neural Network.

1 Introduction

Machine learning algorithms have been able to perform remarkably in a huge variety of applications like: web search, spam detection, caption generation, speech and image recognition. However these traditional algorithms often fail to make accurate predictions for a model when there is very little amount of information available. Siamese Neural Networks(for image recognition) are able to perform accurately under such restrictions. Consider the figure below, a SNN works by getting input of pairs/triplets of images (Anchor,Positive and Negative) and then calculates the distance between positive and anchor as well negative and anchor. During the training process it tries to reduce the distance between the positive and anchor and also increases the distance between anchor and negative to create a new space that groups images that have the same label together.

Therefore the choice of Sampling is of utmost importance for such a network. Our focus is to understand the impact of Sampling in performance of such network.

Figure 1: Learning discriminative features

2 Related work

The triplet loss technique has been widely used in several computer vision applications such as: face recognition, image retrieval and person re-identification. One work for reference is: **FaceNet** for recognition of a face. This method works by combining triplet loss technique with deep convolutional network. In this method an online triplet mining method is used which selects the hardest sample pairs (chap 3) from each batch. A structural loss training method was later added to it which trains the network by lifting the pairwise distances within the batch to the matrix of pairwise distances. A method of batch hard loss with soft margin was used for face identification. This method randomly sampled few instances to create small sets as an input batch and selected some of the hardest instances to calculate the loss.Dong and Shen

Our motto is to enhance the understanding of the impact of sampling methods in a Siamese Neural Network, therefore we have implemented two different kinds of sampling techniques which are: Random Sampling and Balanced Batch Sampling.

3 Methodology

Sampling Strategies
One of the biggest problems before training is the selection of the pairs/triplets from the dataset, especially for the triplets. There are three different types of negative samples to the anchor and corresponding positive sample which are: hard negative, semi-hard negative and easy negative. Figure 2 illustrates these. In this figure, the positive and anchor image are the two white circles (a=anchor, p=positive). The three bigger coloured areas are the boundaries between the three different negative sample types. The red circle represents the area of all hard negative. This area is a circular space around the anchor with a radius that represents the distance between the anchor and the positive sample. This means that all hard negative samples have a smaller distance to the anchor than the positive. The orange space represents the semi-hard negatives. These are the negatives that have a larger distance to the anchor than the positive, but are still in a certain margin. This means that the radius of this circular area is the distance between the positive and the anchor plus a margin. All of the other negatives are easy negatives. Moindrot

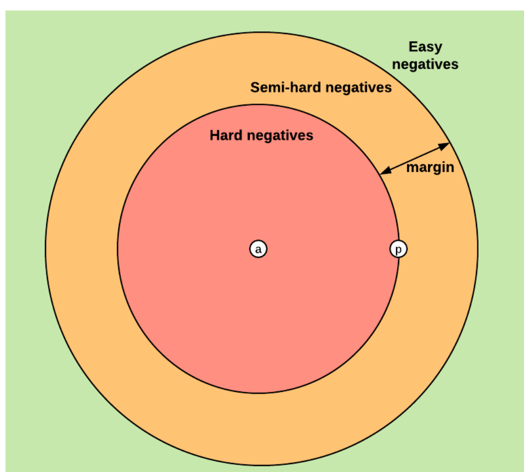


Figure 2: Three different types of negatives, given an anchor and a positive Moindrot

There are also two different ways to mine these pairs/triplets. The first option is offline mining. This strategy computes all embeddings (triplets/pairs) on the dataset and selects all pairs/triplets before each epoch. Moindrot The second way is online mining. In this strategy all useful embeddings are calculated for each batch of images. This strategy produces more triplets/pairs for a single batch than the offline mining. Moindrot

In this report we implemented one triplet and one pair net and used different sampling strategies to train them. As an offline mining method we chose the random selection for both methods. This means that the pairs/triplets are randomly generated before each epoch based on the whole dataset. For the online mining strategies of the pair model, we chose the all-positive and hard-negative. Both mining strategies select all possible positive samples and the same amount of negatives in each batch and only differ in the selection of the negatives. All-positive uses random negative samples and hard-negatives, as the name suggests, chose only the hardest negatives, with the smallest distance to the anchor. For the online mining strategies of the triplet model, we chose an all-triplet-selector, a random-negative-triplet-selector and a semi-hard-negative-triplet-selector. The all-triplet-selector uses all the triplets of all selectors because it generates all possible triplets in each batch. The random-negative-triplet-selector selects a random negative sample and the semi-hard-negative-triplet-selector a semi-hard negative sample for each positive to generate a triplet .

Dataset

Three datasets, are used to compare the different sampling strategies. All of these three datasets consist of 60.000 training and 10.000 test images that are divided into 10 classes. In this report the split of the training and testing was adjusted by removing one class from the training dataset, since it is the purpose of a Siamese neural network to calculate the distance between the images of an unknown class correctly. Due to this the new numbers for training are 54.000 images, containing the classes 1-9, and 6.000, only of the class 0, for testing. The datasets are MNIST LeCun, KMNIST Clanuwat et al. [2018] and Fashion MNIST Xiao et al. [2017]. MNIST is a dataset that classifies 10 different hand written numbers, KMNIST differentiates among 10 different handwritten Hiragana characters and FashionMNIST contains clothing images of 10 different clothing types.

Code

Libraries used: pytorch, torchvision, numpy, matplotlib, PIL, argparse and itertools. This network is based on the work of Bielski et al.. Changes were made in loss function, sampling strategies and trainer. The code for the accuracy, dataset loader, inference and the printouts of the results are newly created.[1]

Loss Function

Contrastive loss was used for all SNN trained with pair sampling and triplet loss for the other networks. The loss function's main purpose is to minimize the similarity metric of two pairs if they are from the same category and maximize it if they are not Chopra et al. [2005]. As a similarity metric it uses the L2-Norm for positive pairs and the Hinge Loss for the negative pairs.

$$L(a,b,y) = y * ||a-b||_2 + (1-y) * max(0, m - ||a-b||_2)) \tag{1}$$

where (a,b) is the image pair, $y \in \{1 = \text{positive pair}, 0 = \text{negative pair}\}$ and m is the margin.

While the contrastive loss only uses either the positive or the negative pair, the triplet loss takes both into account Medela and Picón [2019].

$$L(a,p,n) = max(0, ||a-p||_2 + m - ||a-n||_2) \tag{2}$$

where (a,p) is the positive pair and (a,n) is the corresponding negative pair.

Accuracy

To calculate the accuracy for the triplet model we used the proportion of the number of triplets in which the feature distance of the positive pair is less than that between the negative pair.

$$\frac{\sum(||a-n||_2 - ||a-p||_2)}{c} \tag{3}$$

where c is the number of triplet. To obtain similar accuracy for the pair models we divided the number of pairs by two and used this number as c, because the validation set has the same amount of positive and negative pairs. Since the pair models do not have positive and negative pairs at the same time, the target value y from the loss function is needed again. Following is the formula after the adjustments for the pair model:

$$\frac{\sum((1-y)||a-b||_2 - y * ||a-b||_2)}{c} \tag{4}$$

[1] Available at: https://github.com/LVBreuer/SampleSelectionSiameseNN

4 Results

All of the models were trained on an NVIDIA 2070 super GPU, an AMD Ryzen 5 5600x CPU, a Samsung 980 CIe 3.0 SSD and a Tomahawk II Motherboard with 16GB main memory.

All datasets were divided into batches of 128. The models are then trained with a learning rate of 0.001 with Adam as an optimizer. The learning rate was reduced by a multiplicative factor of 0.1 every 8 epochs.

We trained all of the seven sampling strategies with each of the three datasets to get a direct comparison among these methods under different circumstances. During the training there was nearly no difference among the datasets, but a lot among the sampling selection itself. The first notable difference was in the training time. As Table 1 illustrates, the all positive online triplet selection model needed far more time than any other trained model. The reason for this can be found in right column of the same table. This method computed 200 times more samples in each batch than the maximum number of samples for the other models. Surprisingly, the online pair selection models were the fastest during training although they computed more samples than the random offline pair selection model.

	MNIST		KMNIST		FashionMNIST		Sampling Pairs
	Train	Test	Train	Test	Train	Test	
Random Offline Pair	256	**87**	256	88	225	84	128
Hard Negative Online Pair	**141**	89	142	**81**	**142**	91	2.700
All Positive Online Pair	142	85	**141**	86	**142**	82	2.700
Random Offline Triplet	350	91	347	94	346	89	128
All Positive Online Triplet	2677	89	2759	90	2664	**80**	540.000
Random Negative Online Triplet	687	**87**	716	84	782	83	2.700-1
Semi-hard Negative Online Triplet	679	96	691	90	676	91	2.700-1

Table 1: Training time in second, testing time in milliseconds and sampling pairs per batch.

The next notable difference was the training loss. All of the models had a similar low loss except for the random negative online triplet and the semi-hard negative online triplet selection. For each dataset these two models improved their loss during the training, but it started and ended a lot higher than the others. In comparison to the training loss, these two models did not produce a significantly lower accuracy, as can be seen in Table 2 during training and testing. The first small difference among the three datasets is visible in the training loss and accuracy as well. Since nearly every selection method produced low loss and high accuracy for the MNIST dataset, the other two had two methods with an accuracy below 70 percent. The hard negative online pair and random negative online triplet selection for KMNIST and the random negative online triplet and semi-hard negative online triplet selection for FashionMNIST did not train properly based on the training loss and accuracy.

We used two different evaluations for the validation of the models. Since all the models were trained with datasets containing nine different classes, the models were tested with only these nine classes first and then the missing class was added later to test the models on an unknown class. The accuracy values for all models reduced for both testing strategies, but the test-set with only nine classes produced a better accuracy than the other one.

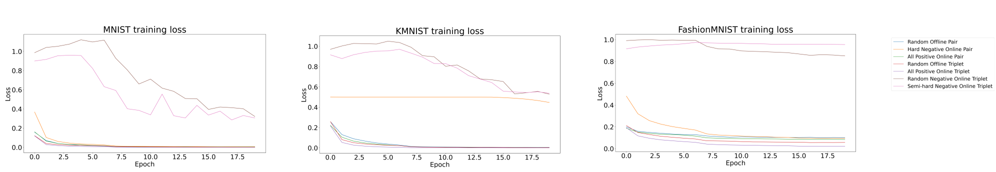


Figure 3: Training loss for datasets.

The results of all the training and testing mentioned above indicates that nearly all models learned to separate known classes from each other. But most of the models had problems to separate an unknown class from the known. Figure4 illustrates the results of a model before and after the training and it shows that before the training all classes lie on top of each other and after the training they were separated. Figures 4 and 5 represent the difference between the datasets and the classification based on the hard-pair and all-triplet selection models. The hard-pair selection model of FashionMNSIT has a low accuracy in training and testing for this dataset and the result shows that there is no clear class separation, even between the trained classes. In contrast to this the validation results of the all-triplet selection for the FashionMNIST clearly seperates the unknown (red) class from the others. Even tough the MNIST dataset produced a better accuracy, the graphical results of the all-triplet selections (middle plot of Figure 5) show a better seperation of the unknown class than the MNIST result (middle plot of Figure 4). We also noticed that during training it is very important to check how similar the unknown class is to the existing classes. The right image in Figure 5 shows the results of a model where the unknown class was the T-shirt and in the middle the Trouser was unknown. The model trained without the T-Shirt class matched all T-Shirts to the existing class Shirts and the model trained without Trousers grouped them together as a separate class.

	MNIST			KMNIST			FashionMNIST		
	Test10	Test9	Train	Test10	Test9	Train	Test10	Test9	Train
Random Offline Pair	**98.89**	99.25	99.87	**98.89**	**99.03**	**99.97**	**97.89**	**98.14**	**99.57**
Hard Negative Online Pair	94.19	99.25	99.73	33.66	42.56	74.66	80.54	87.66	99.70
All Positive Online Pair	98.28	**99.72**	95.07	95.25	98.19	99.89	95.39	95.81	99.92
Random Offline Triplet	97.75	99.54	**99.92**	93.49	97.18	99.92	95.73	96.65	99.93
All Positive Online Triplet	97.33	99.64	99.58	93.66	97.31	99.87	96.04	97.26	99.98
Random Negative Online Triplet	77.83	82.23	93.78	66.37	74.48	83.67	56.24	60.23	93.65
Semi-hard Negative Online Triplet	85.79	89.99	99.53	80.90	91.79	99.82	67.65	76.60	95.55

Table 2: Training and Testing accuracy in percent. Test10 is the test with all ten classes and Test9 only with the 9 trained classes.

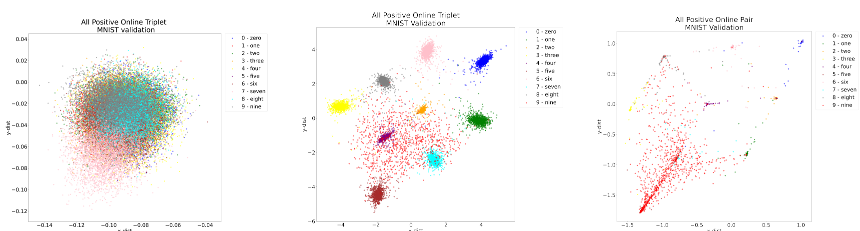


Figure 4: Results left before training and middle/right after training

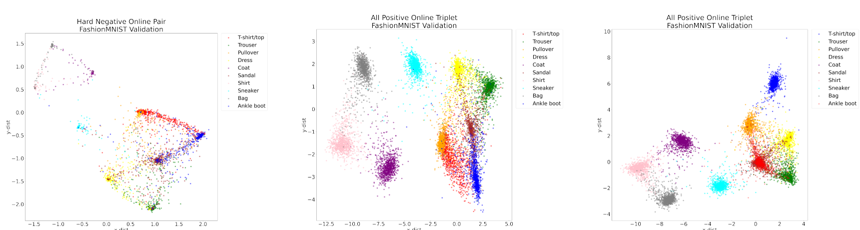


Figure 5: Results of the all triplet/pair selection methods FashionMNIST validation dataset. The red class is the untrained class.

5 Conclusion

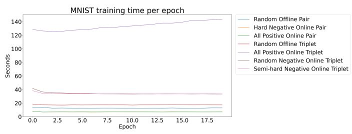
In this paper we investigated the impact of various sampling techniques in a Siamese Neural Network for its overall performance. We noticed, that if a method like All-Positive-Online-Triplet, takes the largest amount of time to train, and also processes the maximum number of samples, it does not necessarily mean it will have the highest or lowest accuracy for all the input datasets when compared to another method like All-Positive-Online-Pair.

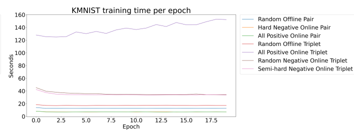
We can hence deduce that a single sampling technique can not guarantee us the desired result in every case, therefore the choice of sampling technique is quite significant. We might be able to close this gap by expanding the scale of convolutional neural networks.

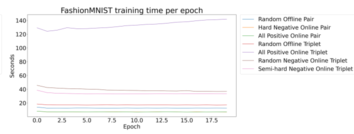
6 Appendix

These are all results for all sampling selection methods on all three datasets.

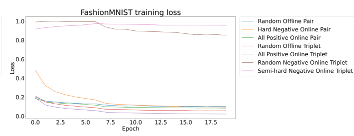
6.1 Training Time

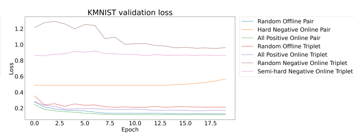


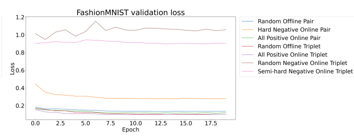




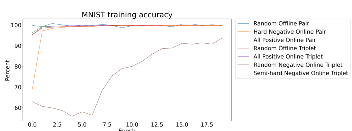
6.2 Loss

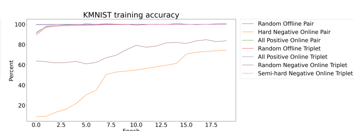


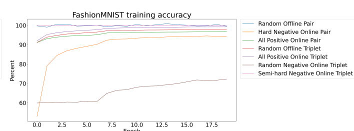


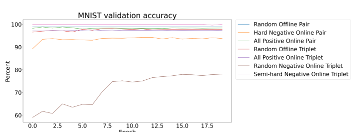


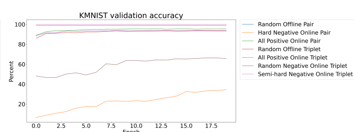
6.3 Accuracy

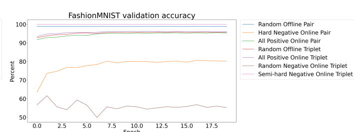




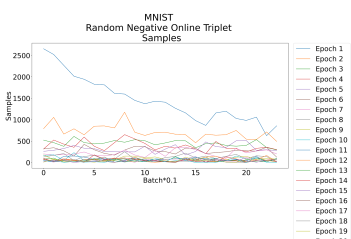


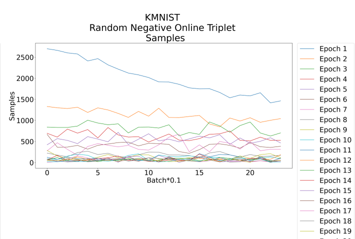


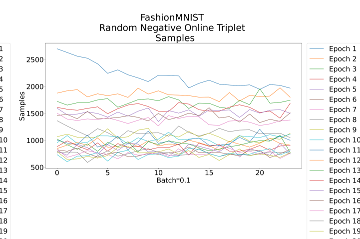


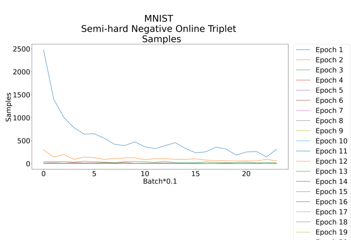


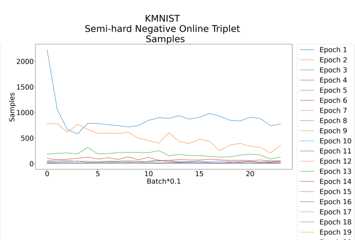
6.4 Sample Selection

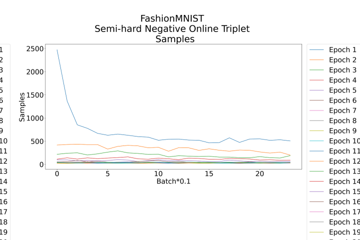








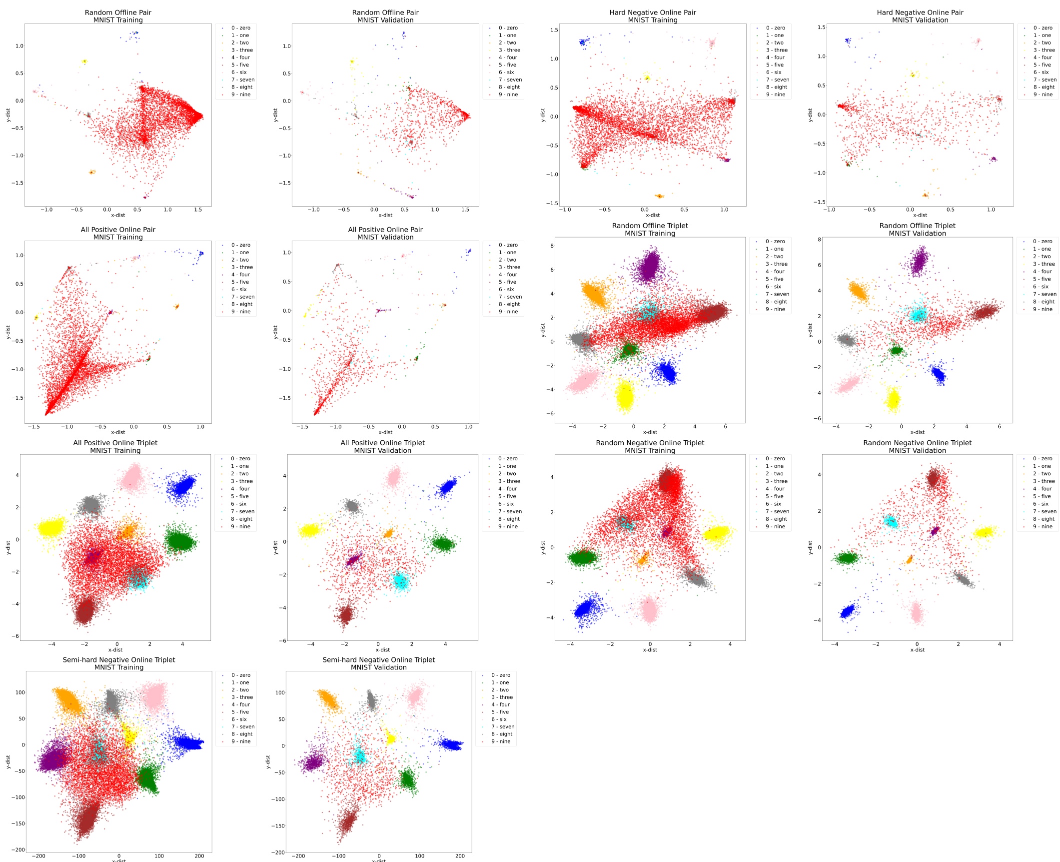




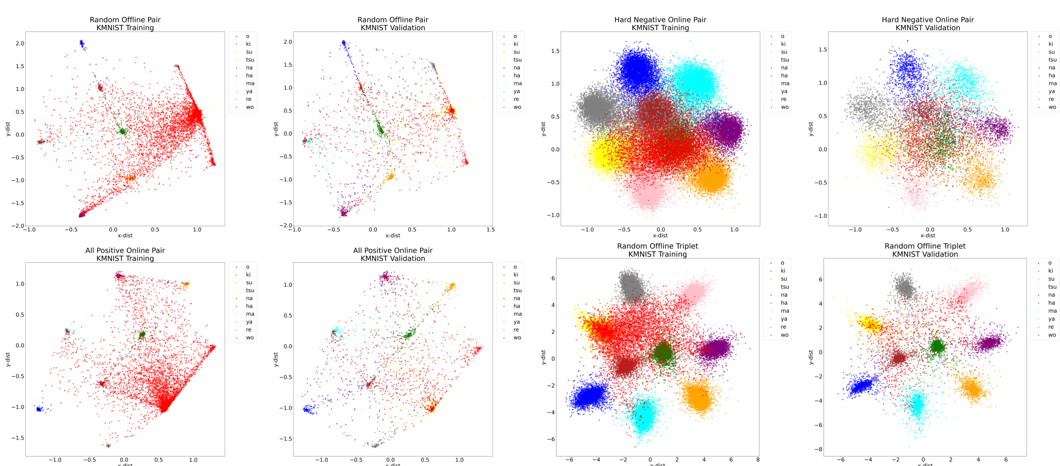
6.5 Results

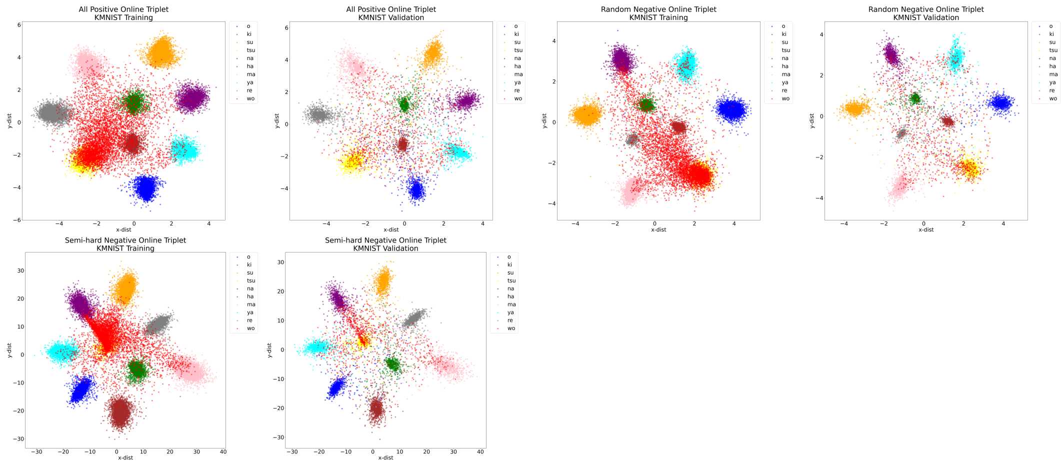
The red class is the unknown class, not inside the training set during the training, in each image.

MNIST

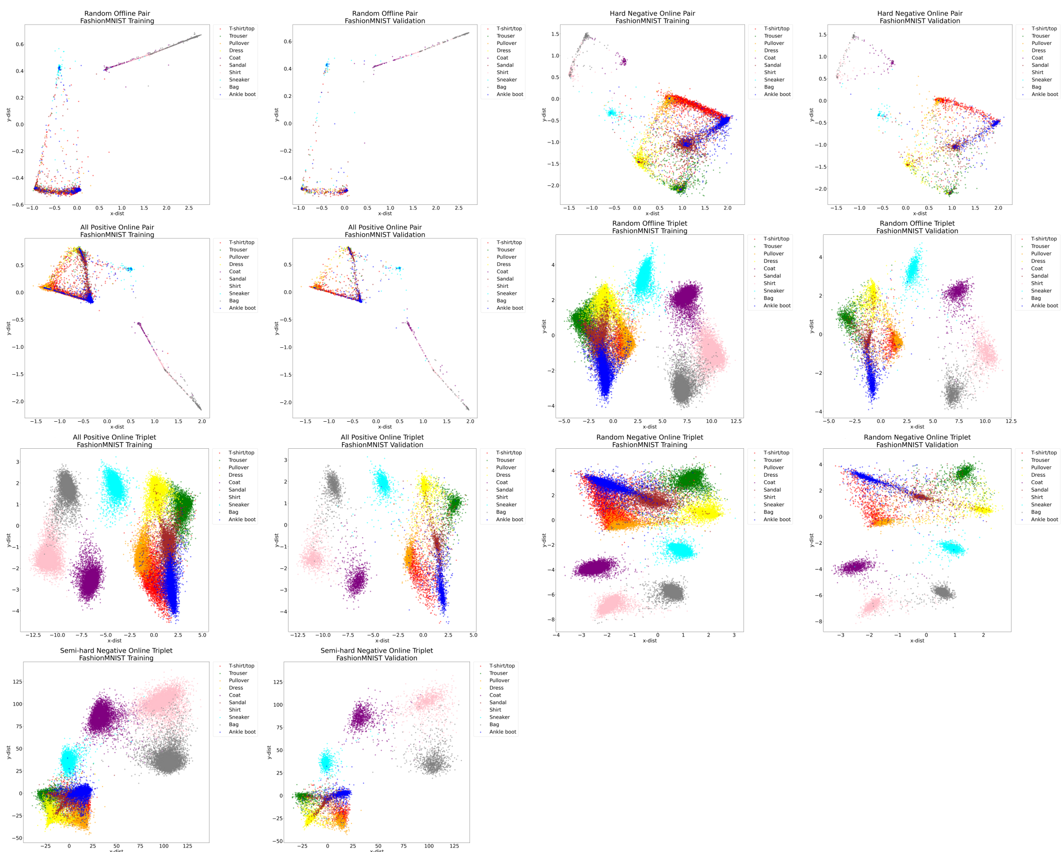


KMNIST





FashionMNIST



References

A. Bielski, K. Tarplee, and Aleksey. Siamese and triplet learning with online pair/triplet mining. URL `https://github.com/adambielski/siamese-triplet`.

S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, 2005. doi: 10.1109/CVPR.2005.202.

T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. Deep learning for classical japanese literature, 2018.

X. Dong and J. Shen. Triplet loss in siamese network for object tracking. URL https://openaccess.thecvf.com/content_ECCV_2018/papers/Xingping_Dong_Triplet_Loss_with_ECCV_2018_paper.pdf.

Y. LeCun. The mnist database of handwritten digits. URL http://yann.lecun.com/exdb/mnist/.

A. Medela and A. Picón. Constellation loss: Improving the efficiency of deep metric learning loss functions for optimal embedding. *CoRR*, abs/1905.10675, 2019. URL http://arxiv.org/abs/1905.10675.

O. Moindrot. Triplet loss and online triplet mining in tensorflow. URL https://omoindrot.github.io/triplet-loss.

H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.