AML Ex2

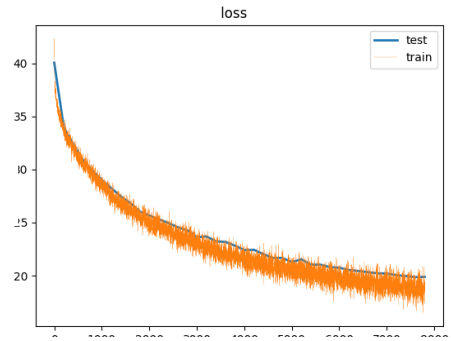
Amitai Ovadia 312244254

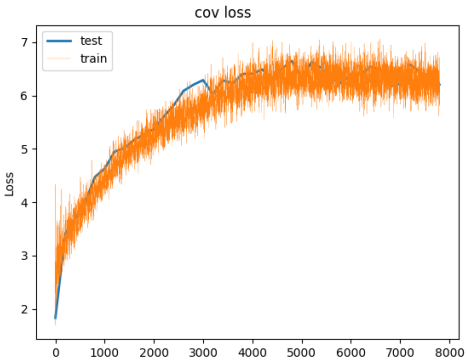
1. Training:

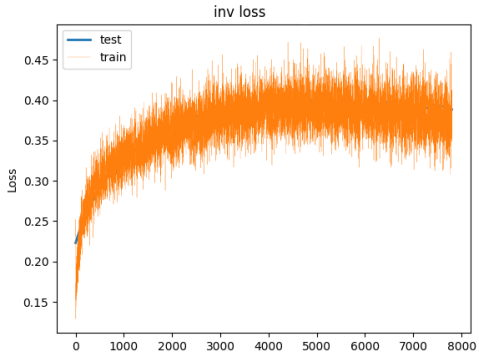
   One the right there is the overall loss term.
   The loss was composed of the term:

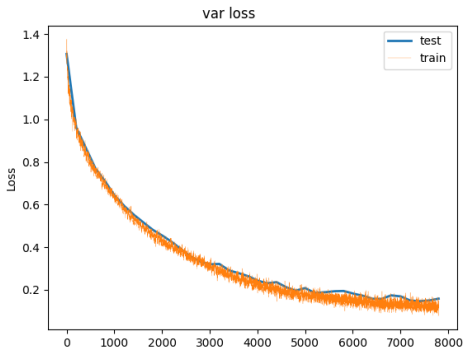
$$\mathcal{L}_{VICReg}(Z, Z') = \lambda \cdot \mathcal{L}_{inv}(Z, Z') + \mu \cdot [\mathcal{L}_{var}(Z) + \mathcal{L}_{var}(Z')] + \nu \cdot [\mathcal{L}_{cov}(Z) + \mathcal{L}_{cov}(Z')]$$

   So here are the other loss components without the λ, μ, ν terms.
   The x axis is the number of batches.



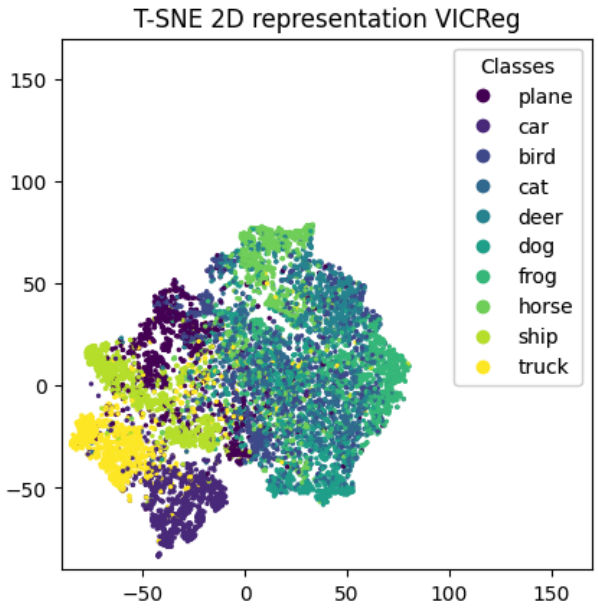


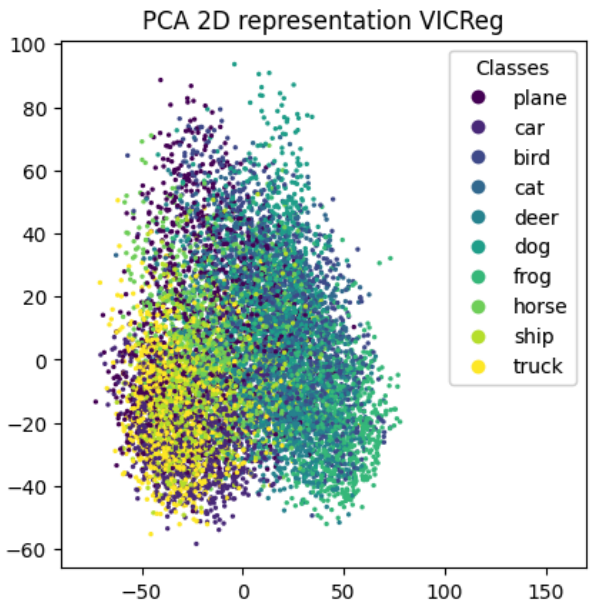




   It's interesting to note that the inv loss and the covariance loss are increasing.

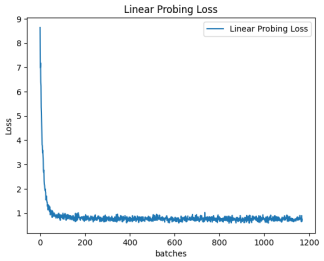
2. For this question I calculated the representations of the test set samples using the encoder part of the network. Then I computed the PCA and T-SNE 2D representations of them.



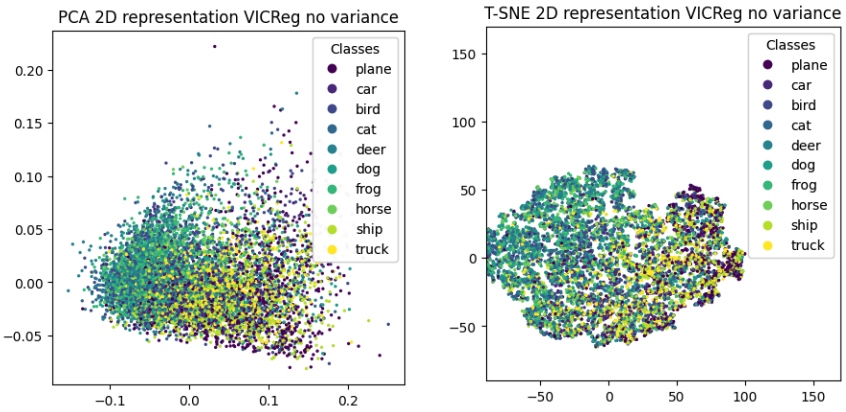


It's clear from the results that the T-SNE has managed to cluster the results in 2D better than the PCA did, as it is much less entangled. We can also see that the track, car ship and plane representations are much better clustered than the other ones, for example the deer, horse, dog and frog representations are very much entangled and hard to separate. Probably because the images are much more similar and indistinguishable in the pixels domain as well.
It's hard to tell if VICReg managed to capture the class information accurately, it's true for some of the classes, the non animals classes, but given the 2D representations it seems as though the animals classes are still hard to separate (we will see in the linear probing part)

3. Here we train a linear classifier on top of the encoded representations over the training set. In the architecture I loaded a 'frozen' encoder, added to it a linear layer (from dim 128 to 10). I trained it on the training set for 6 epochs and then evaluated it on the testing set.
   The training loss is presented here on the right.
   In terms of accuracy, the Linear probing achieved 72% on the test set.



4. Now reformulating the loss to exclude the variance term, I trained the model again for 10 epochs (took less time to converge).
   Here are the PCA and T-SNE 2D representations of the data. It's clear that the data is much more entangled and hard to separate. Basically meaning that the representations that were learned are worse than the ones learned with the variance term.



The reason for that might be that the variance term regulates the demand that each vector z_i in the batch is as different as possible from the vector z_j. And so it forces the encoder to learn representations that are much more spread out in the R^d dimension of the representations, and therefore more separable clusterable.

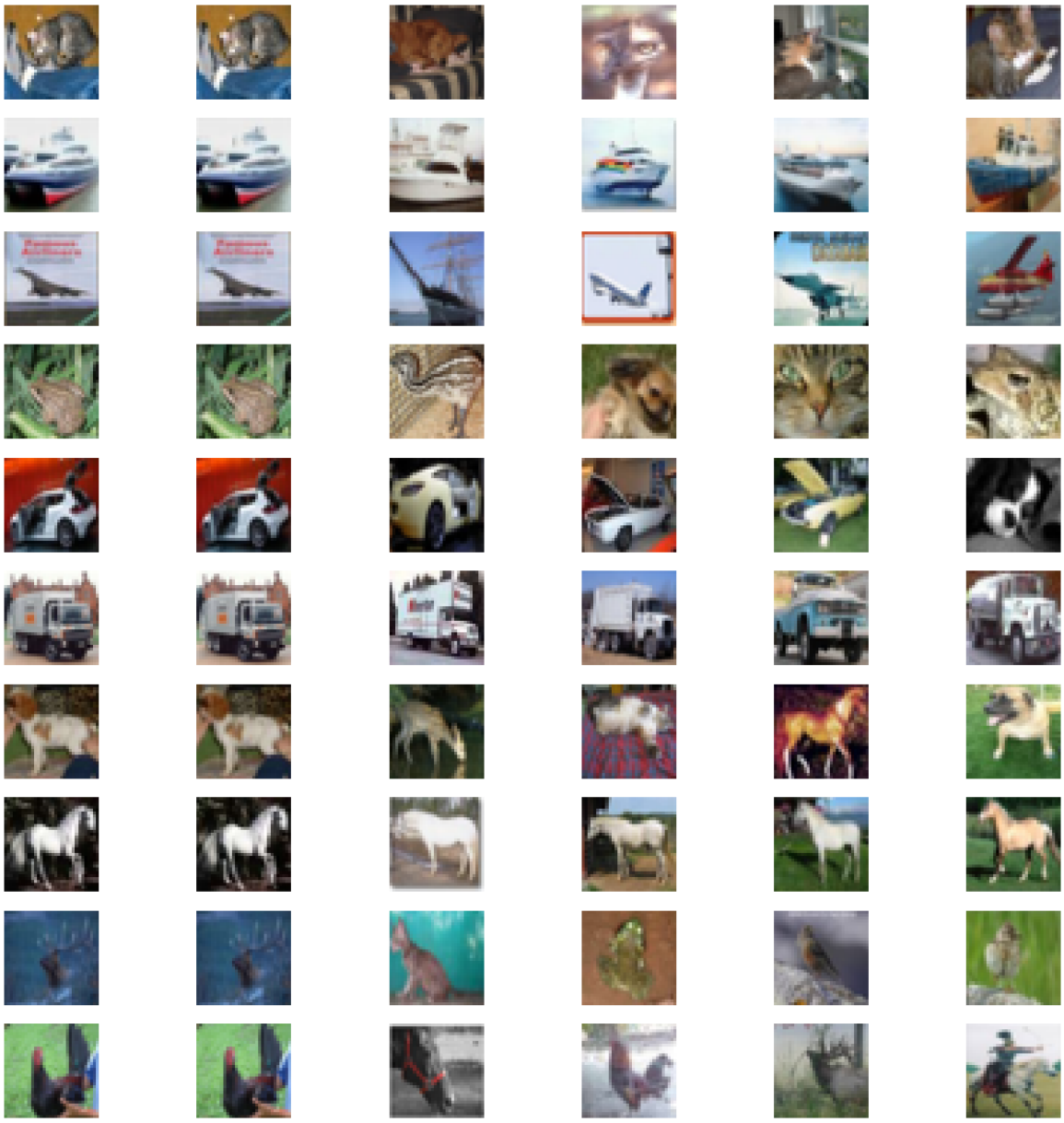
It's getting clearer when we check the ability of a linear classifier to classify the sample using their representations, and the accuracy was just 17% (not far from random).
The visualizations also spread more light about the inability of a linear model to classify between the classes. The PCA, which is also a linear reprojection of the data into 2D failed spectacularly to do any clustering of the data (also the T-SNE) and so it is less surprising to see it failing the classification task.

5. I am not sure I understood the question. How can we replace the encoder with latent optimization? Doesn't that require a trained encoder as well?

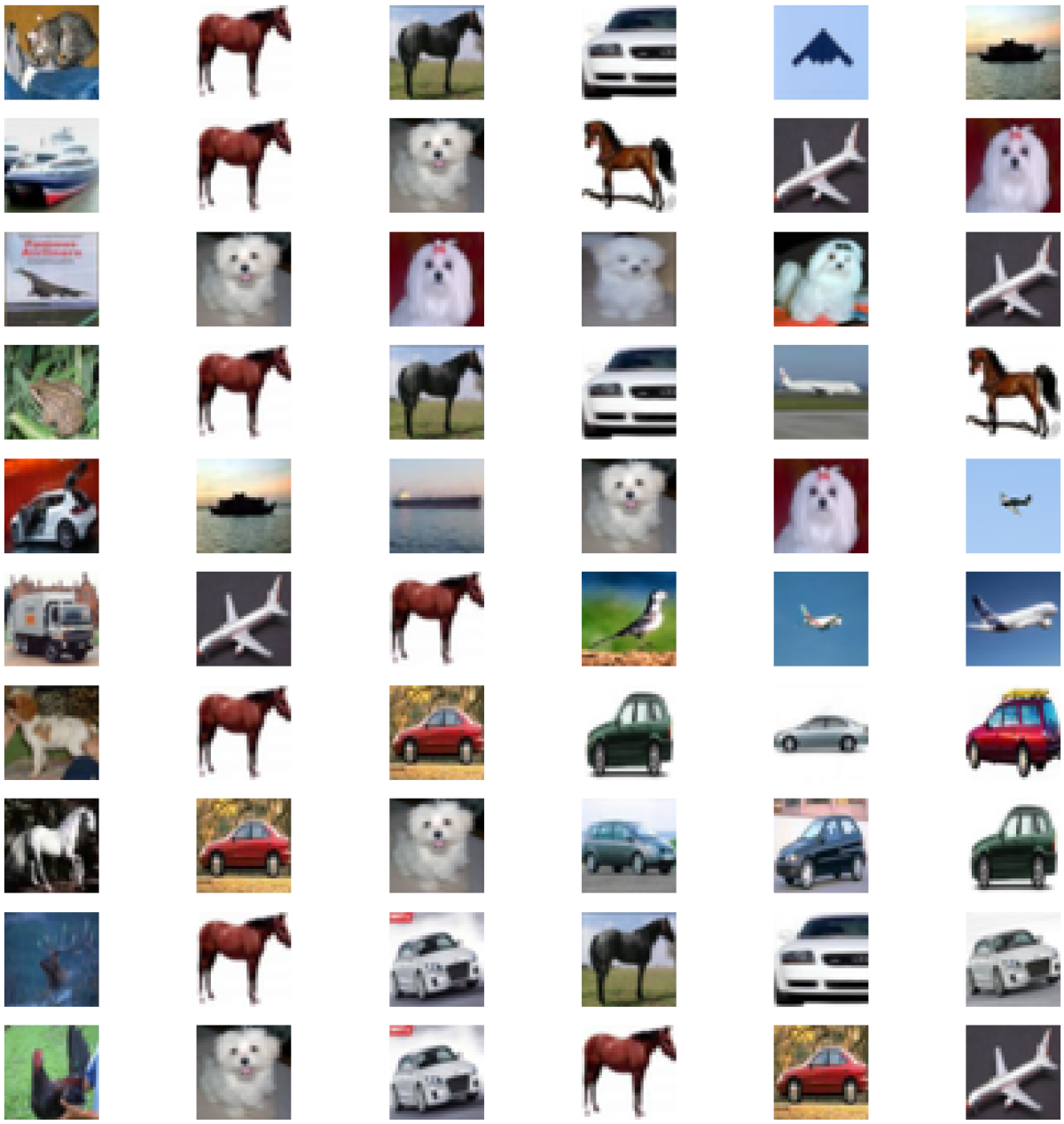
6. I created this new model that was trained such that given img1, then img2 is given as one of its 3 nearest neighbors in the latent representation trained in Q1.
   The accuracy of this model after applying to it a linear classifier was:
   ```
   Accuracy of the network on the test images: 32 %
   ```
   Which is worse than the original model, but better than the model in question 4.
   A reason for that might be that we don't ensure during training that img2 is semantically close to img1.
   An added value that this algorithm might have is that the resulting augmentations in Q3 change the image less, and in a more light way then taking an other image from a similar class and enforce their representations to be close, it might be a more robust way to represent the data.

7. It looks as though the laplacian eigenmaps representation learning ON cifar10 is a total disaster for downstream classification tasks compared to the VICReg's T-SNE plot from Q2. Based on both the accuracy level (the same accuracy as random) and the visual representation, it seems that the representations that were learned by the laplacian eigenmaps (by definition) try to preserve the same distances between images in the pixel domain.
   This is of course a totally useless quality, as it tells us nothing at all about the semantic content of the images.
   On the other hand, VICReg's representation learning paradigm tries to learn representations that are close to other representations by their semantic similarity (by enforcing similarities between 2 augmented images representations of the same image).
   And so it's clear why VICReg worked better than the laplacian eigenmaps in our task of representation learning.

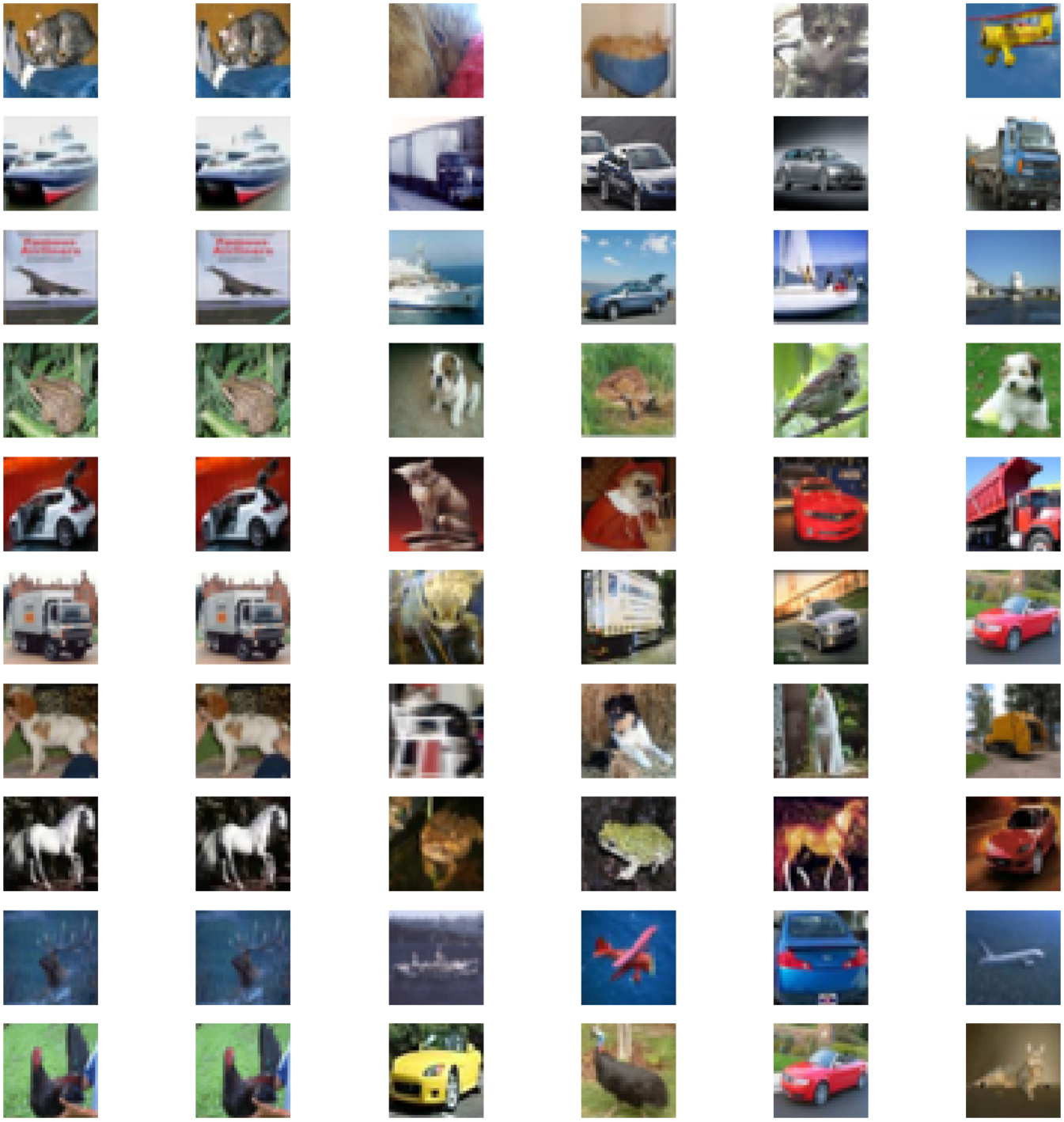
8. Here are the closest neighbors of the model from Q1:
   The original image is the most left one



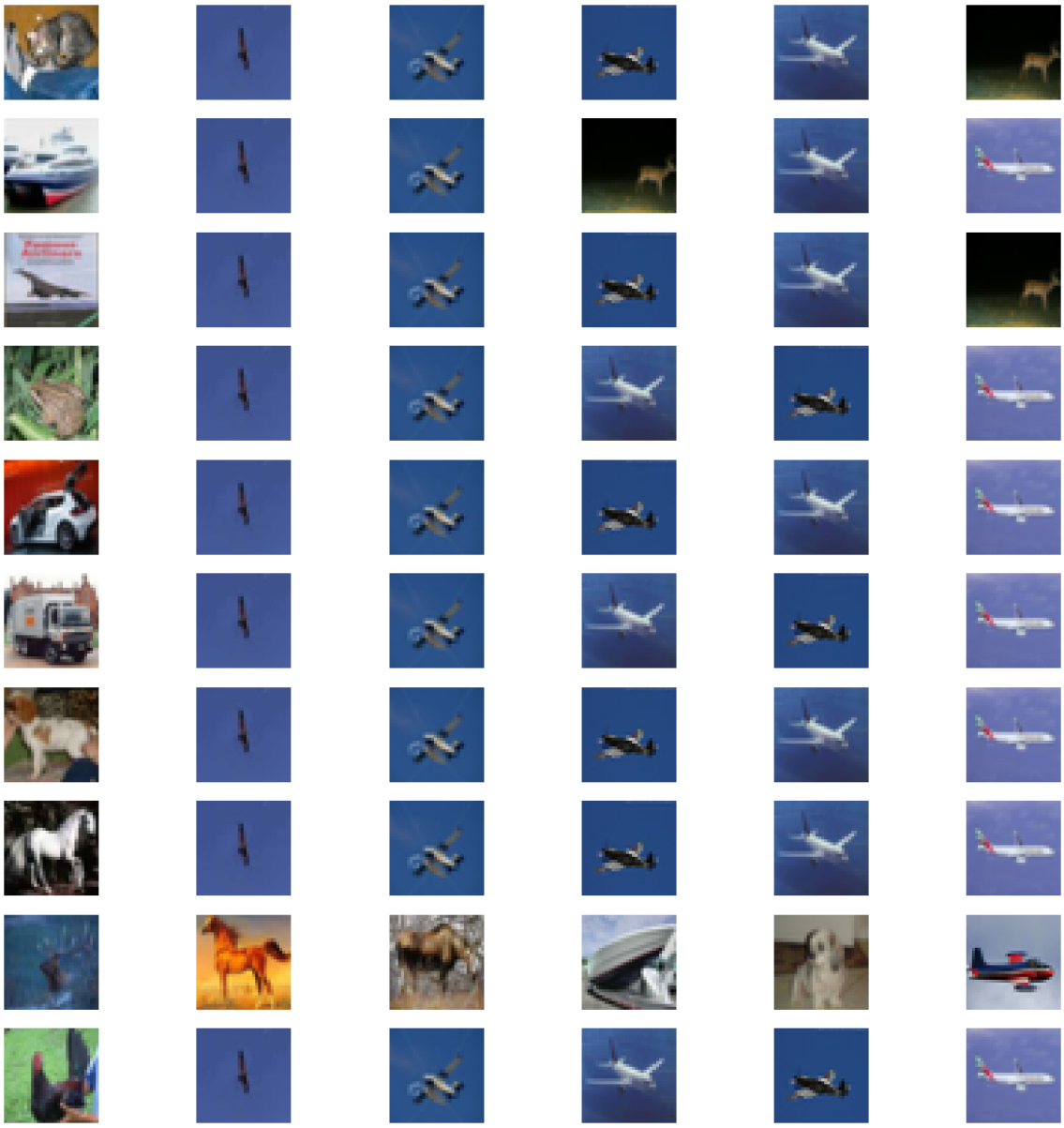
This are the furthest samples in latent space from each image using the representations of Q1:



Here are the close neighbors of the model from Q6:



This are the furthest samples in latent space from each image using the representations of Q6:



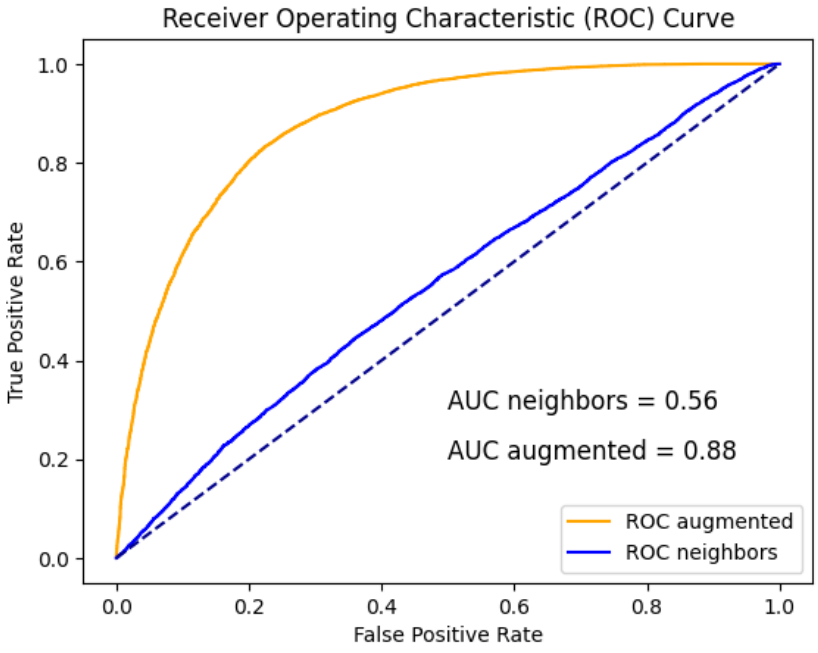
- I feel like the Q6 model has performed pretty badly (in my case) in comparison to the original Q1 model in the area of clustering semantically similar samples together as neighbors. The Q1 model managed, in almost all the cases, to keep images from the same class as neighbors, in other words, it captures the semantics of the images much better than the Q6 model.
  It's very clear also in the furthest neighbors task, that it managed to distance classes that are very far from each other.

- The Q6 model on the other hand seems to be more focused on similarities in the pixel domain for some reason, but not class related features. Also the images that were furthest apart appeared to be the same images across all the classes, and those were images with very strong blue backgrounds. (except of the deer with the blue background)

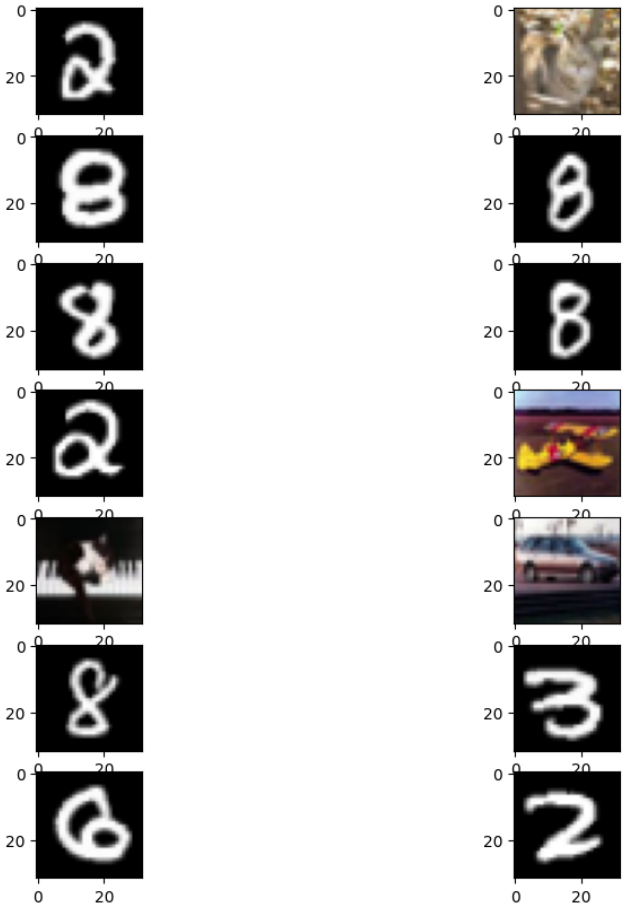
Downstream Applications

1. So we used CIFAR10 as the training data and MNIST data set as the anomalous data. Also the KNN density was defined as the mean distance of sample from the test data to 2 nearest neighbors in the training data.

2. I plotted the ROC curve for the anomaly detection using Q1 and Q6 models.
   As we can see the Q1 model was much better than the Q6 model in the anomaly detection of MNIST data over CIFAR10 data.
   (I know that other people got different results: that the Q6 model was better. I spent a lot of time trying to find out why without success. I hope I won't lose point for it)
   So I will say why the Q6 model should have been better. It was supposed to be more robust because the neighbors of each CIFAR10 sample in the training dataset of Q6 were further apart from one another, so it might have learnt more robust connections between the classes. So the MNIST representations were probably 'pushed' much further in the latent space then the CIFAR10 samples.



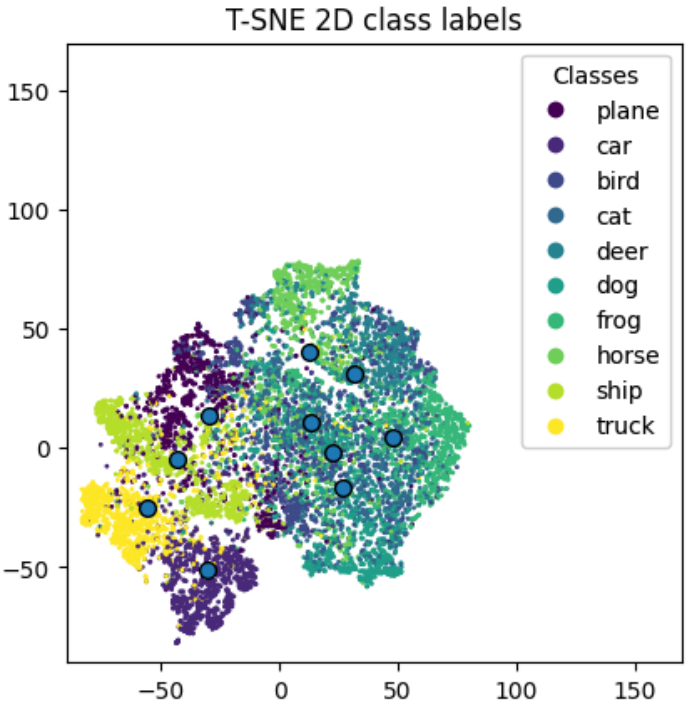
3. In the right side: the Q1 anomalies, and in the left the Q6 anomalies.
   As was stated in the previous section I know those are bad results.
   In my case the Q1 model was much better in recognizing the anomalies but
   I know that other people got different results.
   In my case the Q1 model found the MNIST samples to be anomalies, with an exception of 1 very unrecognizable sample from the CIFAR10.
   In the case of the Q6 model, most of the anomalies were indeed MNIST but it also detected CIFAR10 samples to be anomalies which suggests that something went wrong somewhere on the way.

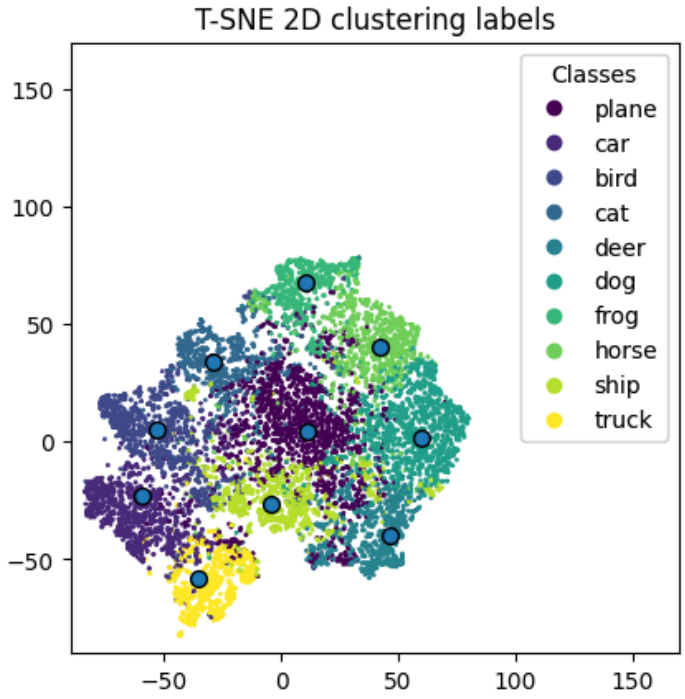


Bonus: clustering

1. I used k means to evaluate 10 clusters for the Q1 model and the Q6 model

2+3. I visualized the T-NSE using the k-means labels and the real class labels.
First for the Q1 model: I also included the centroids of each cluster according to the assigned labels

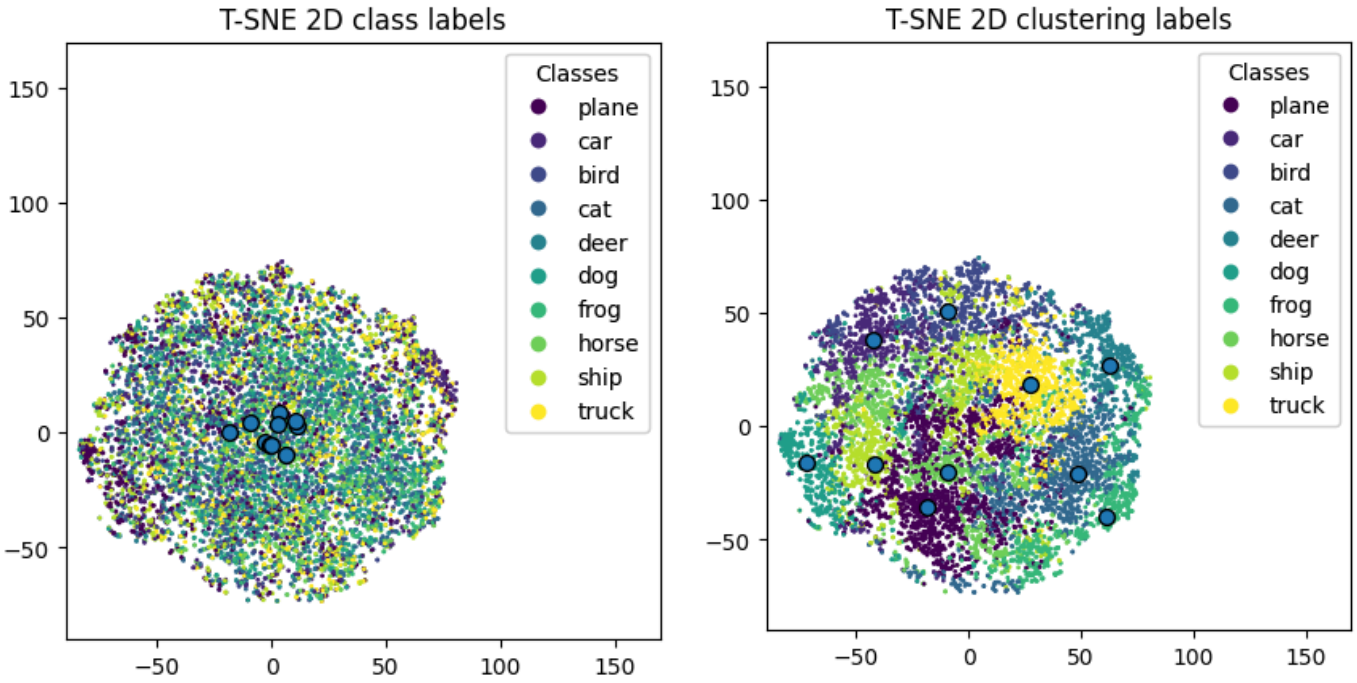




Also it's clear that for the k-means clustering the silhouette score was higher.
The silhouette score is very coherent with the visual analysis: the k-means clustering is much more separable then the original data labels

```
sil score for the clustring labels = 0.1112666130065918
sil score for the real class labels = 0.03042517974972725
```

And now for the Q6 model:




```
sil score for the clustring labels = 0.08936581015586853
sil score for the real class labels = -0.025432275608181953
```

The method that is definitely better in finding clusters is the Q1 model representations. As it can be seen that the centroids of the different classes are much more separable in the Q1 case then in the Q6 case (which are extremely entangled). Also here the silhouette scores are consistent with the visual analysis in the k-means vs the real labels question.