Contrastive Learning

Pattern Analysis and Machine Intelligence

Robin Hammer, Marc Pavlinec, Nathalie Zarbock & Anika Zeilmann

“Contrastive learning is a machine learning technique used to learn the general features of a dataset without labels by teaching the model which data points are similar or different“

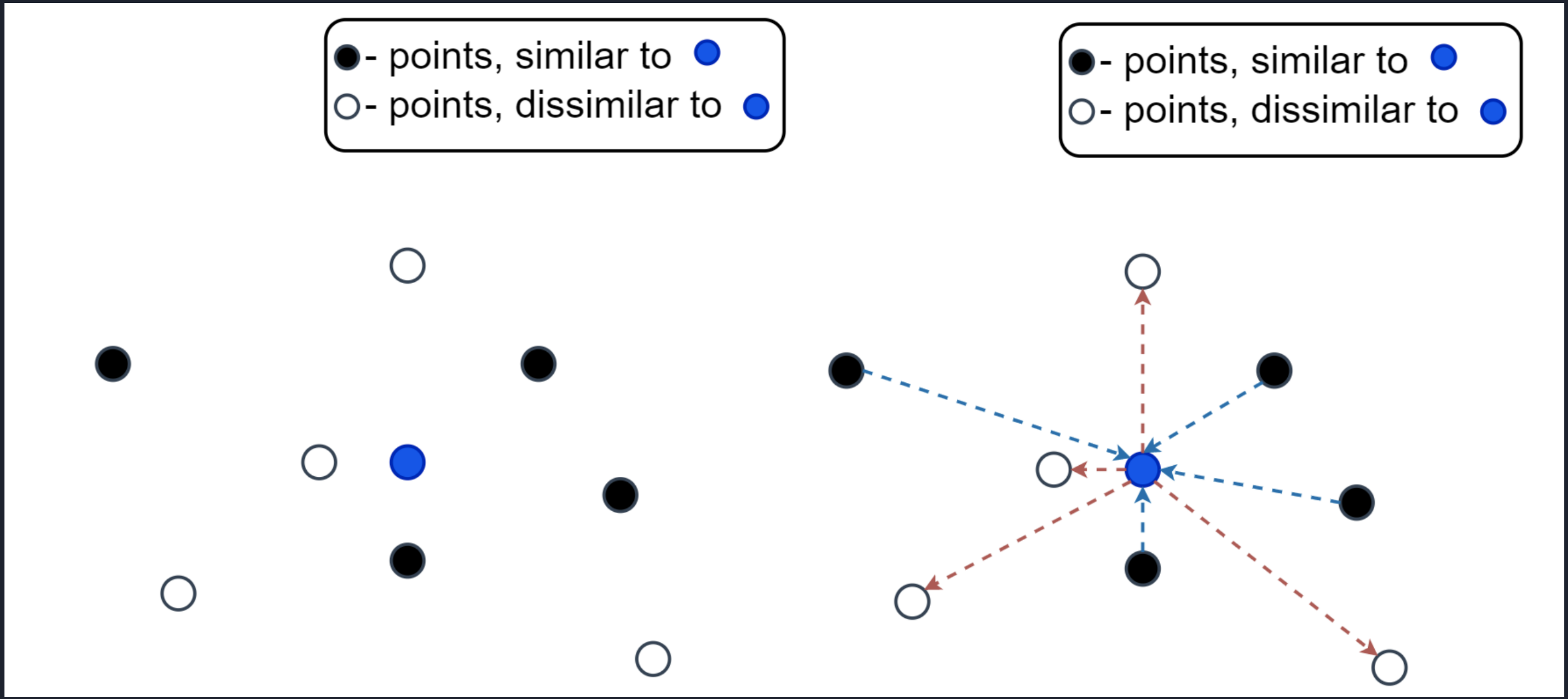
(Tiu, 2021)

Contrastive Learning

- Recognizing the similarities and differences between objects
- Leads to learning the high-level features

Tiu, 2021: https://towardsdatascience.com/understanding-contrastive-learning-d5b19fd96607

Contrastive Learning



Bekuzarov, 2020: https://medium.com/@maksym.bekuzarov/losses-explained-contrastive-loss-f8f57fe32246

Contrastive Learning: 3 Challenges

1. How can we create similar or dissimilar samples?
2. How can we train a network to separate them?
3. How do we evaluate the success?

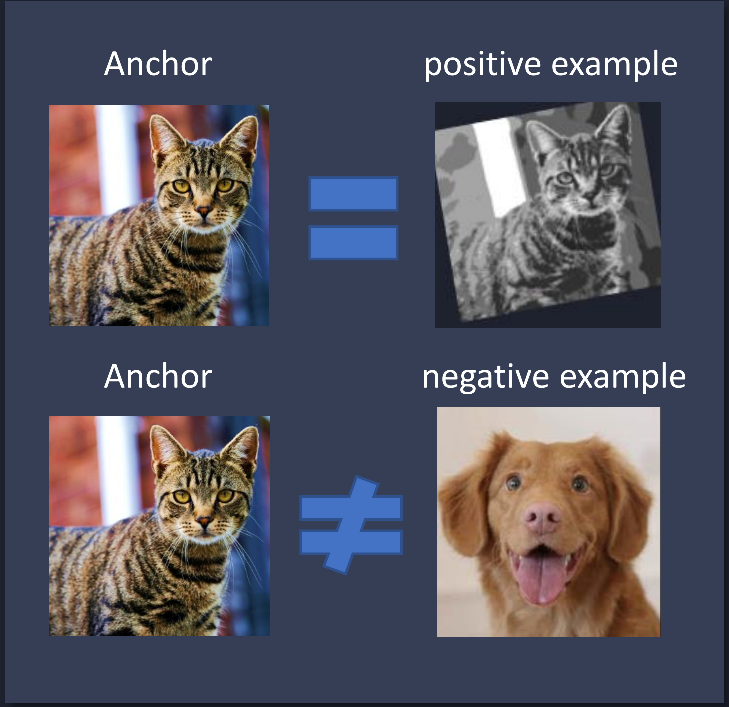
Contrastive Learning: 3 Challenges

1. How can we create similar or dissimilar samples?
2. How can we train a network to separate them?
3. How do we evaluate the success?

How can we create dissimilar or similar samples?

Augmentation

Positive / Negative Pairs



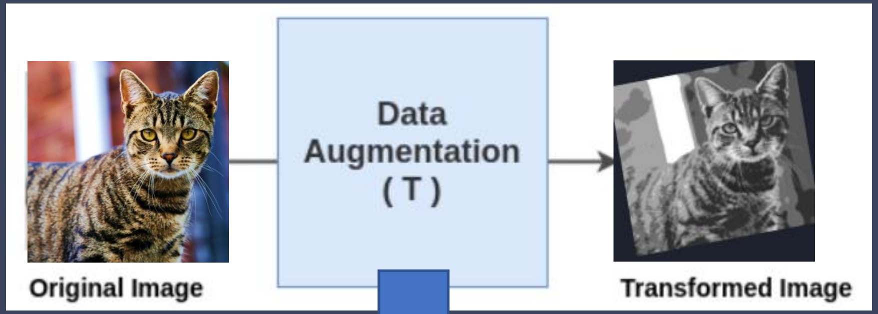
Triplets

Contrastive Learning: 3 Challenges

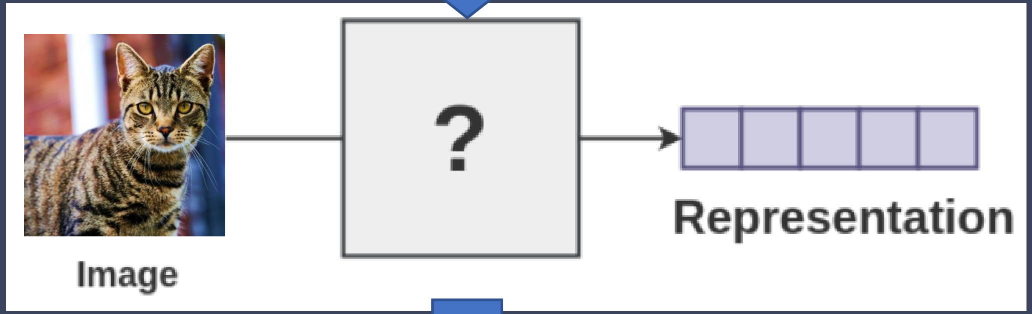
1. How can we create similar or dissimilar samples?
2. How can we train a network to separate them?
3. How do we evaluate the success?

Contrastive Learning

1 Data Augmentation



2 Encoding



3 Loss Minimization of Representations

Tiu, 2021: https://towardsdatascience.com/understanding-contrastive-learning-d5b19fd96607

Contrastive Loss: Cosine Embedding Loss

$$\text{loss}(\text{x}, \text{y}) = \begin{cases} 1 - \cos(x_1, x_2), & \text{if y} = 1 \\ \max(0, \cos(x_1, x_2) - margin), & \text{if y} = \text{-1} \end{cases}$$

Pytorch, n.d.: https://pytorch.org/docs/stable/generated/torch.nn.CosineEmbeddingLoss.html

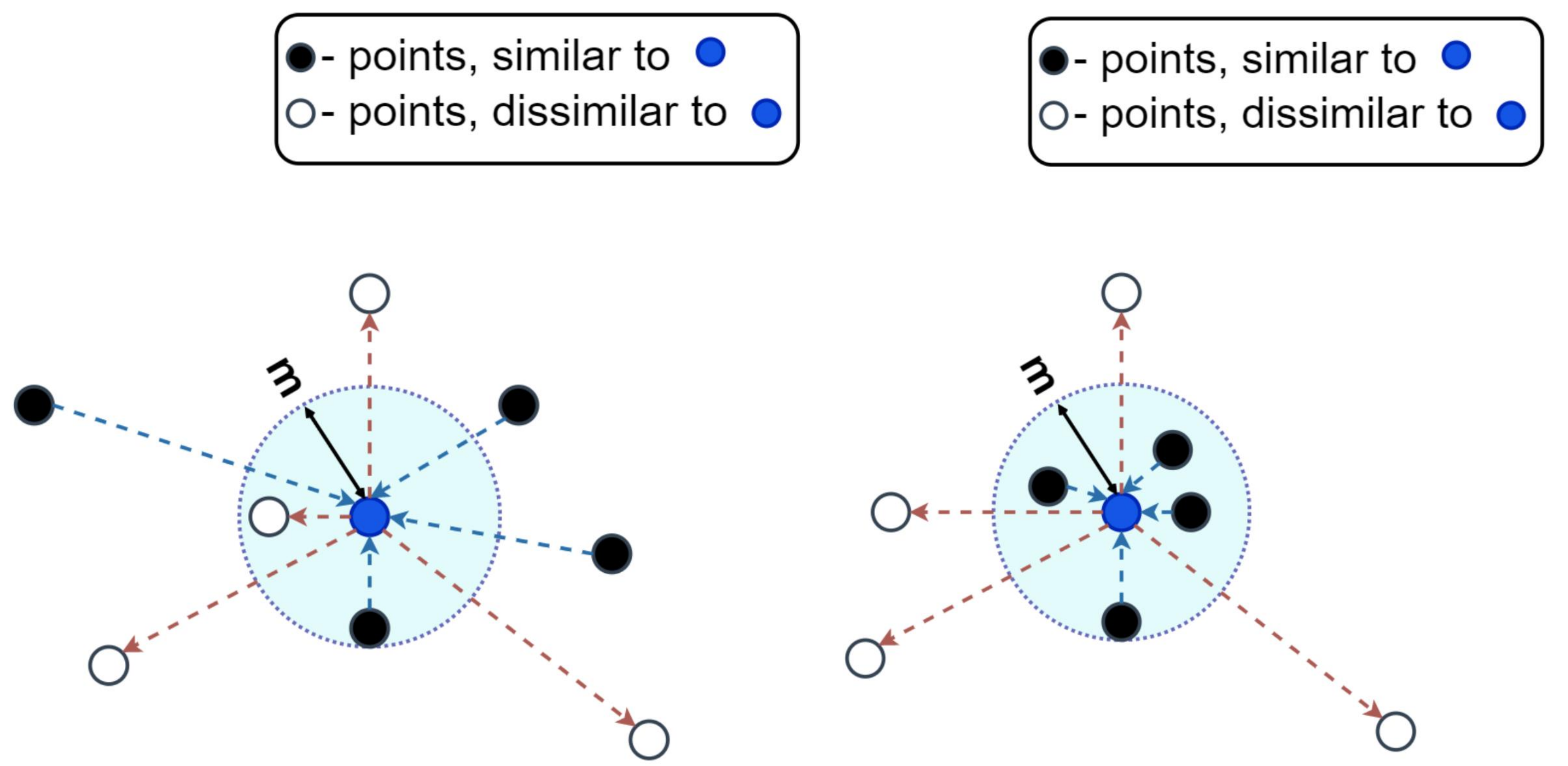
Contrastive Loss: Triplet Margin Loss

$$L(a, p, n) = max\{d(a_i p_i) - d(a_i n_i) + margin, 0\}$$

$$\text{Where} \quad d(x_i y_i) = ||x_i - y_i||_{\mathrm{p}}$$

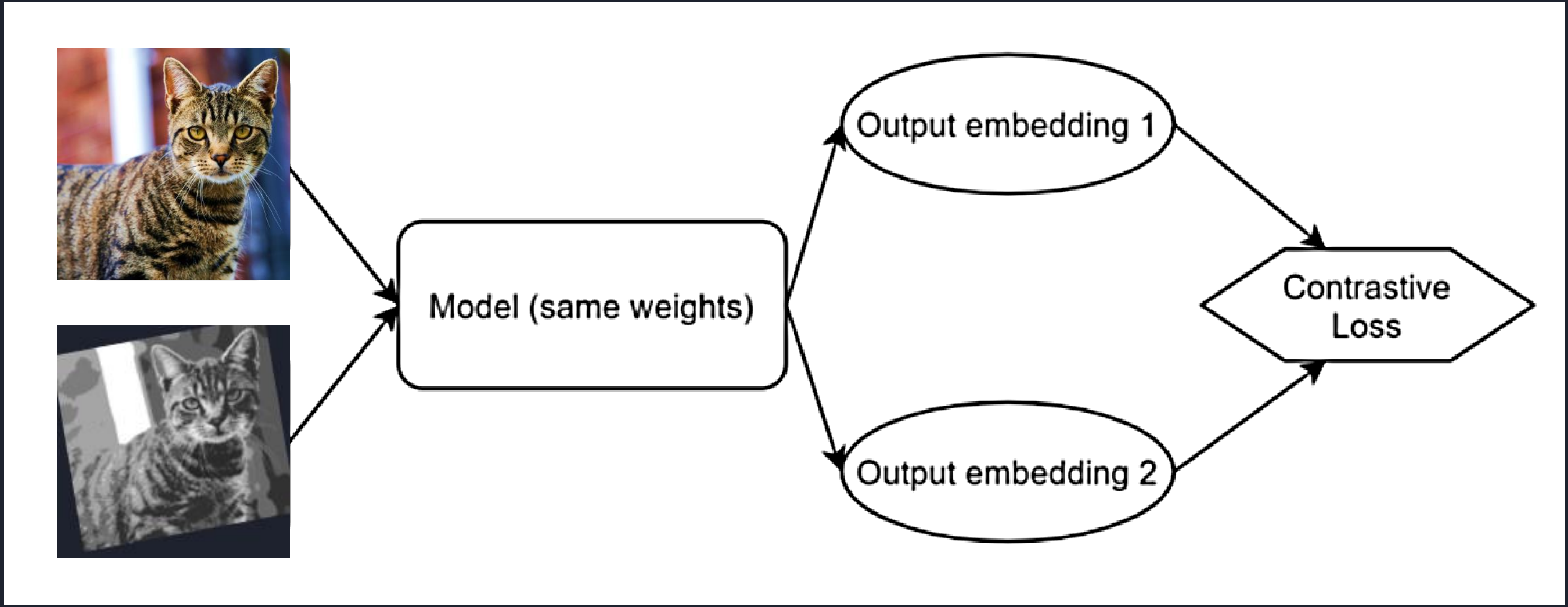
Pytorch, n.d.: https://pytorch.org/docs/stable/generated/torch.nn.TripletMarginLoss.html

Contrastive Learning: Margin



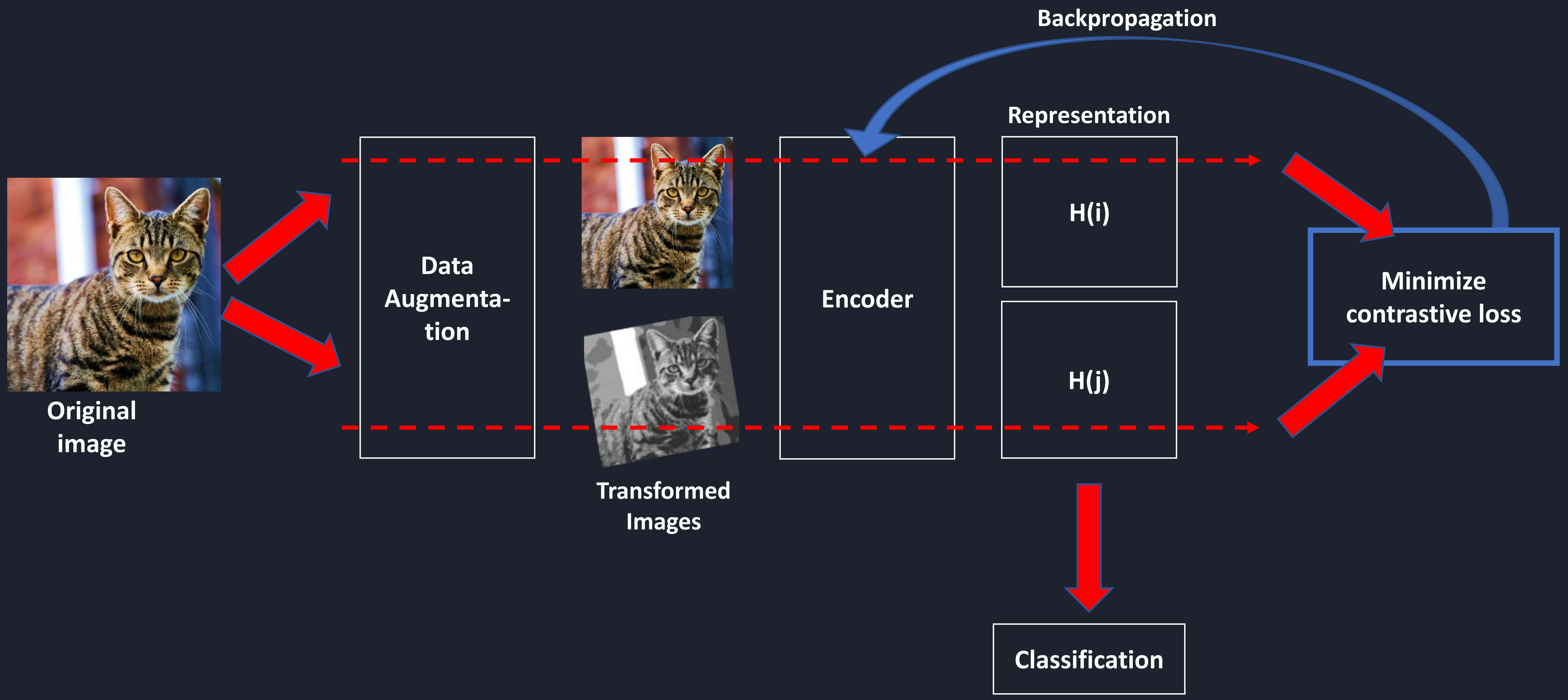
Bekuzarov, 2020: https://medium.com/@maksym.bekuzarov/losses-explained-contrastive-loss-f8f57fe32246

Contrastive learning: Siamese Network Architecture



Bekuzarov, 2020: https://medium.com/@maksym.bekuzarov/losses-explained-contrastive-loss-f8f57fe32246

Experimental Setup: Approach

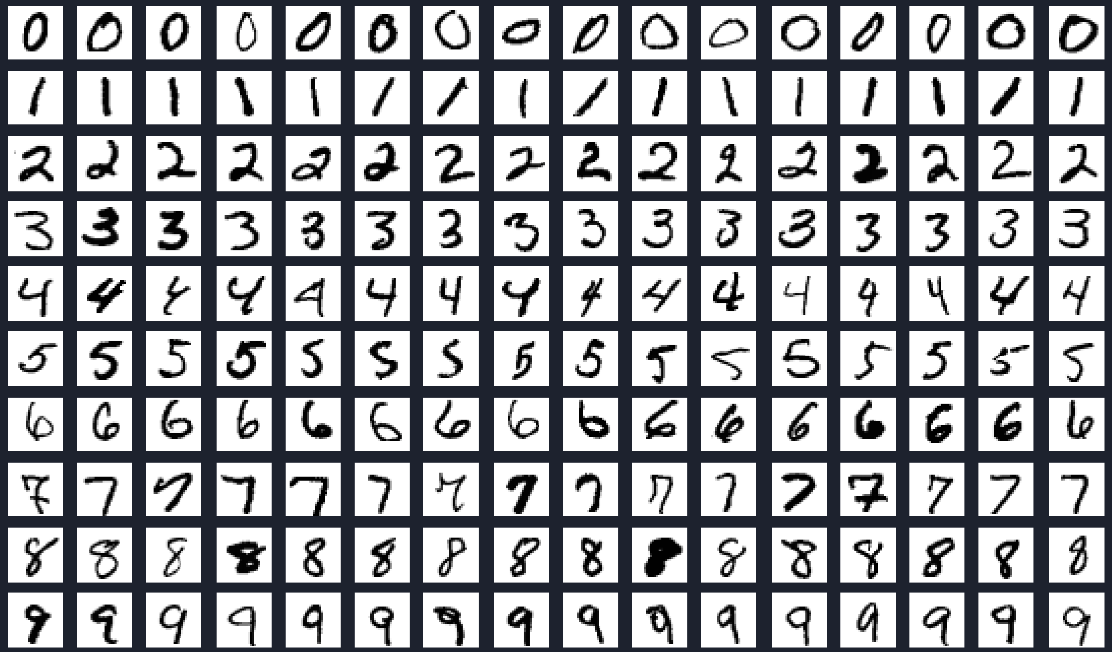


Chaudhary, 2020: https://amitness.com/2020/03/illustrated-simclr/

Experimental Setup: Research Idea

1. Load the data
   - MNIST
   - CIFAR10
2. Use multiple image augmentations
   - Aug 1
   - Aug 2
   - Aug 3
3. Train two CNN:
   - Pairwise
   - Triplet
4. Visualize via t-SNE
5. Train classifier on top of model

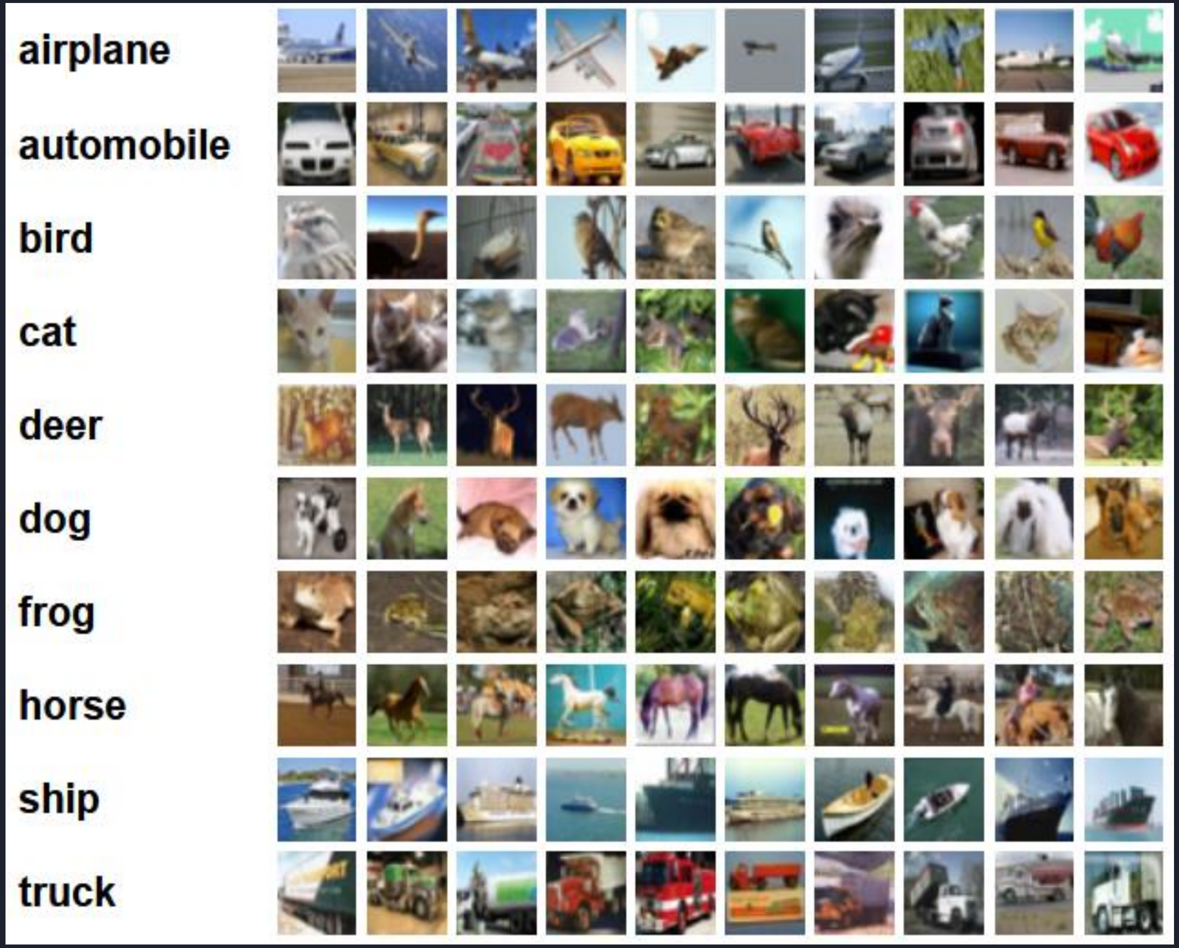
The data set: MNIST



- 10 classes with each class containing 6,000 images
- Classes are mutually exclusive
- 60,000 images for training, 10,000 for testing
- Images are sized 28x28px

(LeCun, Cortes, Burges, n.d.: http://yann.lecun.com/exdb/mnist/)

The data set: CIFAR-10



- 10 classes with each class containing 6,000 images
- Classes are mutually exclusive
- 50,000 for training, 10,000 for testing
- Image are sized 32x32px

(Krizhevsky, n.d.: https://www.cs.toronto.edu/~kriz/cifar.html)

Contrastive Learning: 3 Challenges

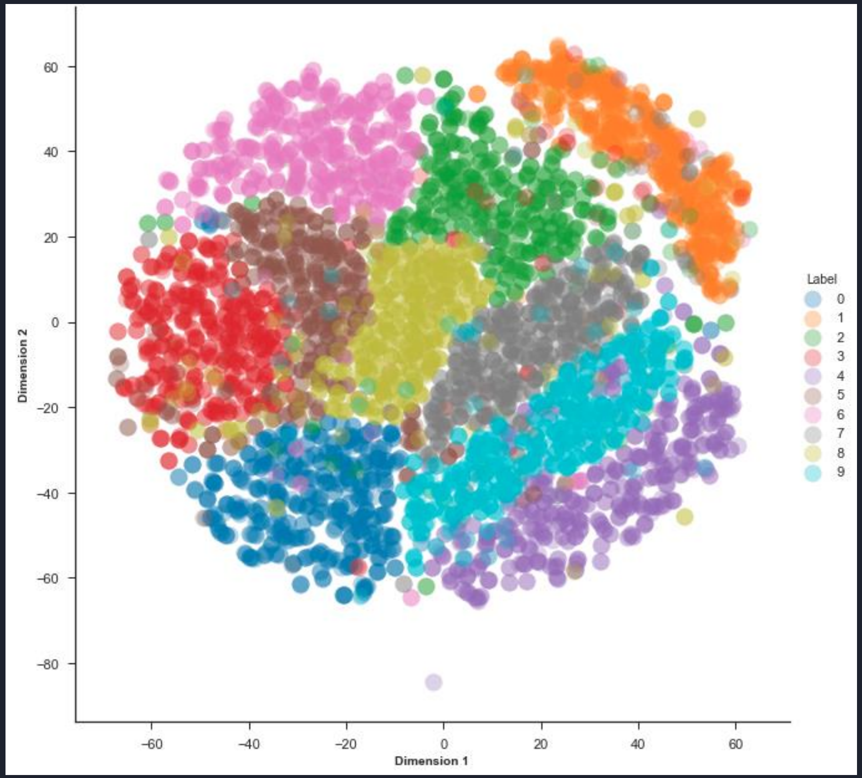
1. How can we create similar or dissimilar samples?
2. How can we train a network to separate them?
3. How do we evaluate the success?

Visualize embedding

Classifier on top of embedding

The Comparison: T-SNE

- Method to visualize high-dimensional data
- Creates intuition on data representation
- Calculates similarity measure between pairs of instances in each high and low dimensional space
- Tries to optimize similarity measures by using a cost function

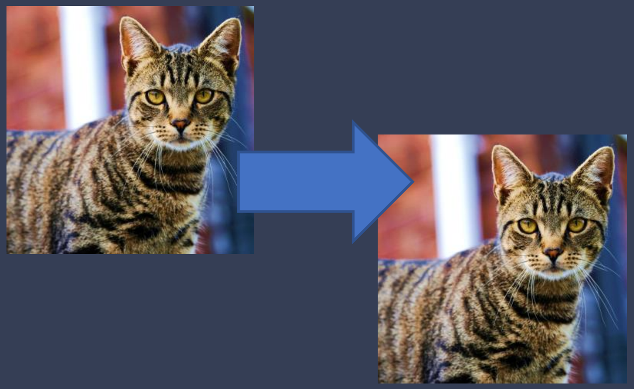


Violante, 2018: https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1

The Comparison: Different Augmentations

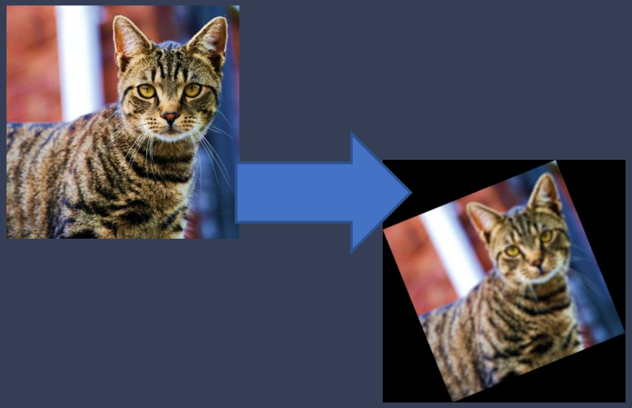
1 None

- Use as baseline for comparing augmentations
- Normalize, conversion to tensor



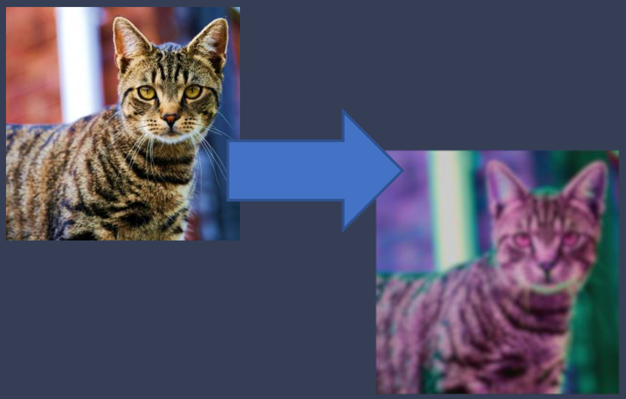
2 Rotation

- Used in *“Deep Neural Networks with Relativity Learning for Facial Expression Recognition”*
- Pad, Rotate and Scale



3 SimCLR

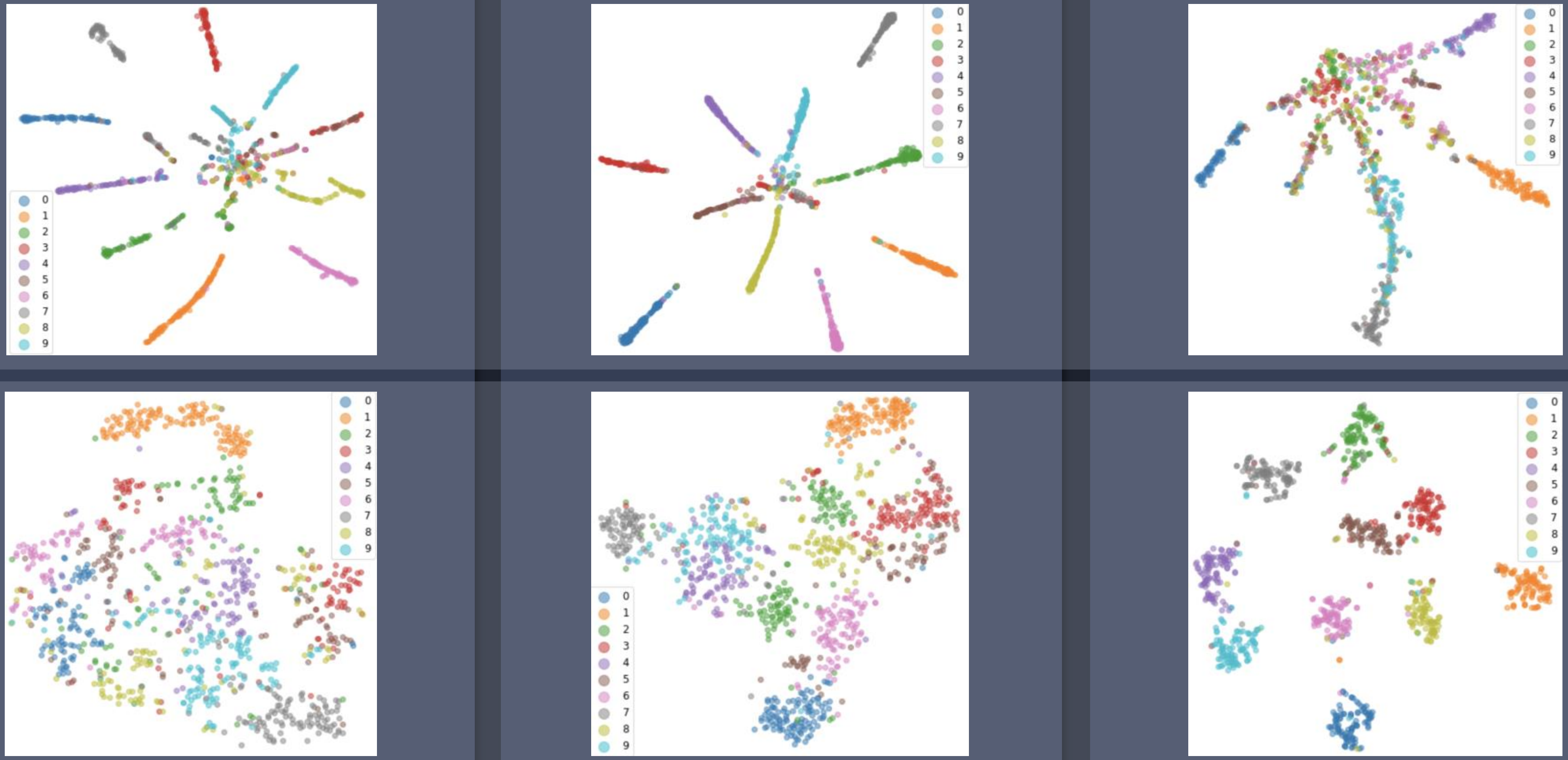
- Used in network *“SimCLR”*
- Augmentation is stochastic
- Random: Crop + Resize, Color Distortion and Gaussian Blur



The Comparison: Different Augmentations

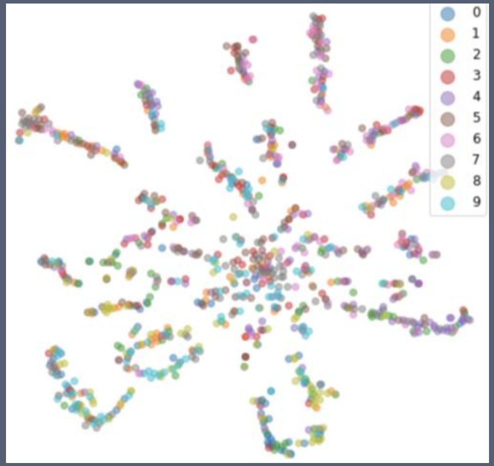
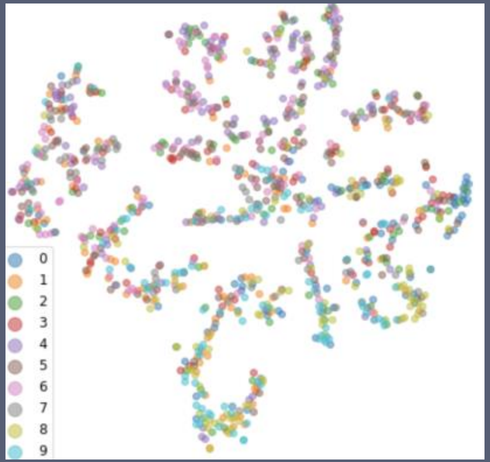
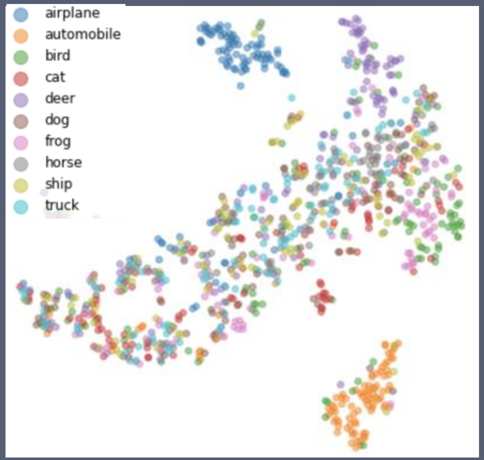
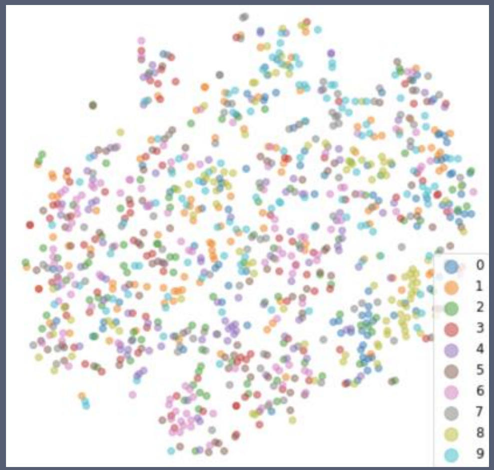
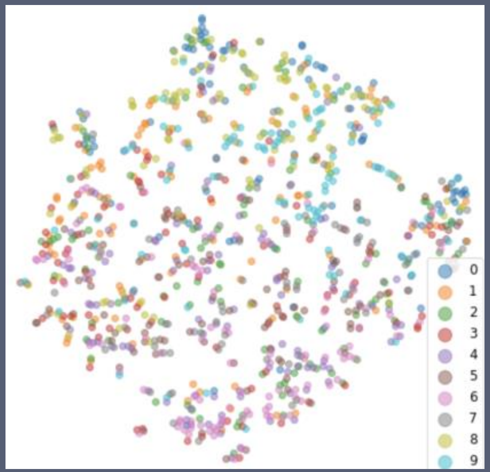
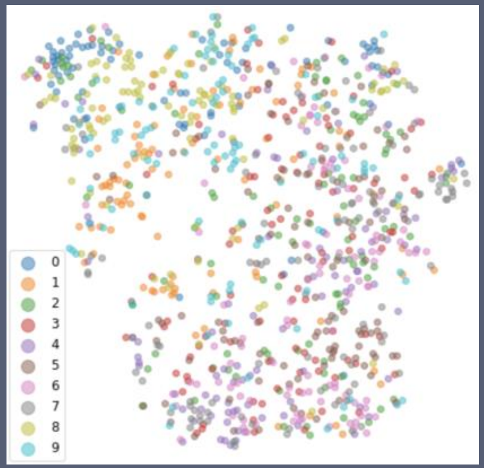
MNIST

	1 None	2 Rotation	3 SimCLR
Pairwise			
Triplet			



The Comparison: Different Augmentations

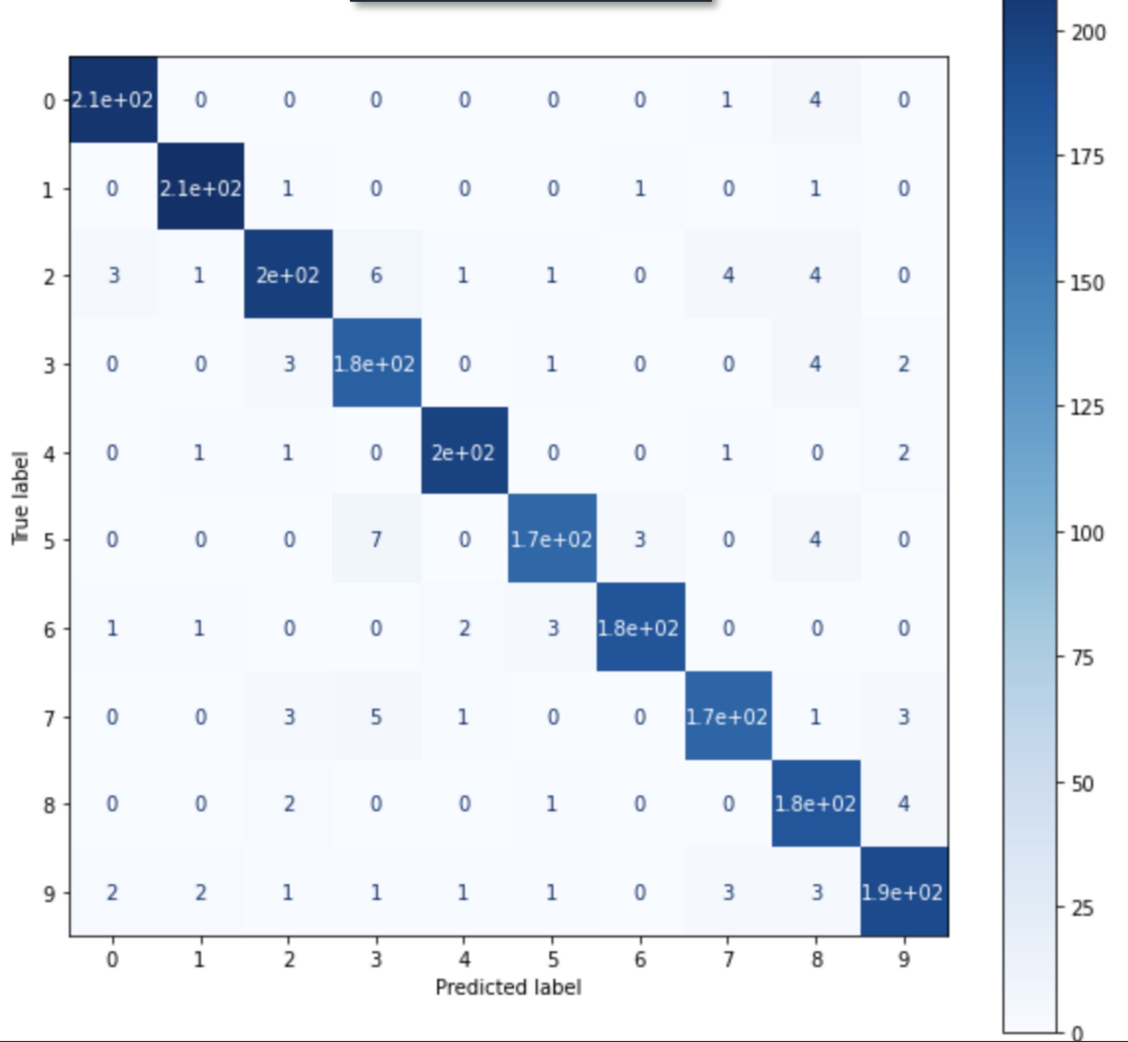
CIFAR-10

	1 None	2 Rotation	3 SimCLR
Pairwise			
Triplet			

Results: Classifier

SimCLR - Triplet

MNIST



CIFAR-10

Observations & Outlook

The augment does impact results, but not necessarily as expected (more is not always better!)

Comparing the data sets, the complexity of pictures seems to influence results

Triplet training led to clustering, whereas pairwise led to line-like structures

Outlook

- Triplet Mining
- Different loss functions
- Supervised contrastive learning

Lessons Learned