MATH6380o Mini-Project 1
Feature Extraction and Transfer Learning on Fashion-MNIST

Jason WU, Peng XU, Nayeon LEE

08.Mar.2018

Introduction: Fashion-MNIST Dataset

- 60,000 training examples and a 10,000 testing examples
- Each example is a 28x28 grayscale image
- 10 classes

index	0	1	2	3	4	5	6	7	8	9
Type	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot

- Zalando et al. intend Fashion-MNIST to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms.

Material: https://github.com/zalandoresearch/fashion-mnist

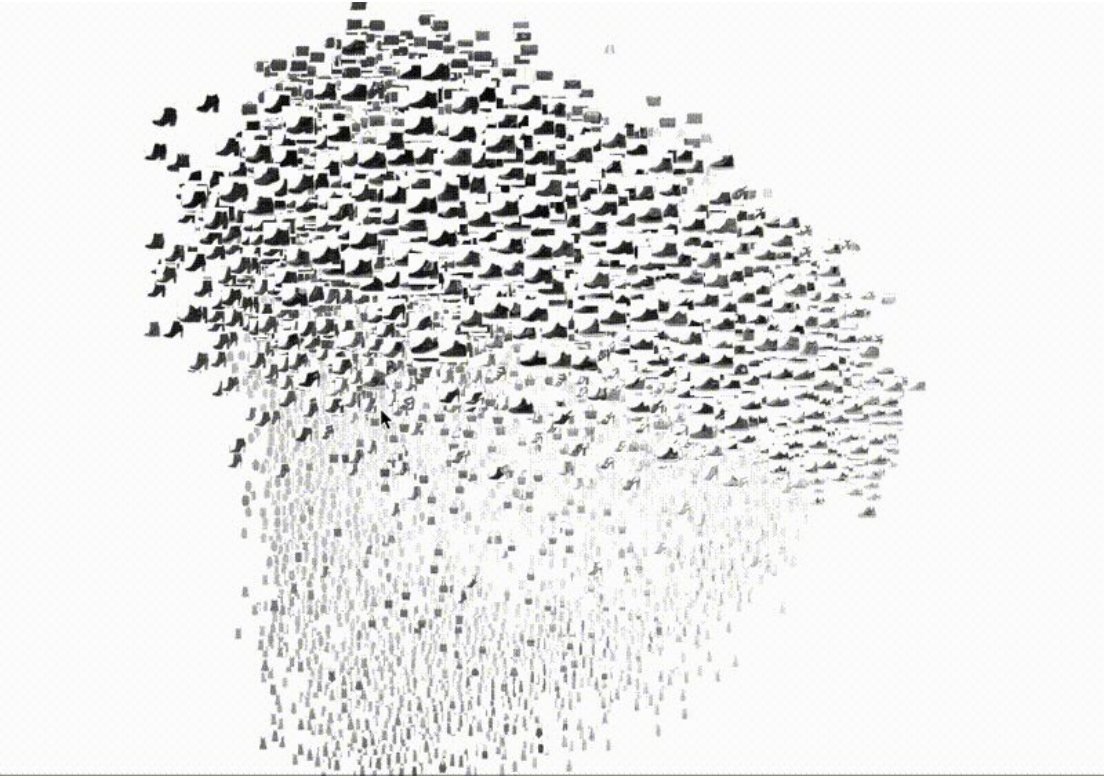
Why Fashion-MNIST?

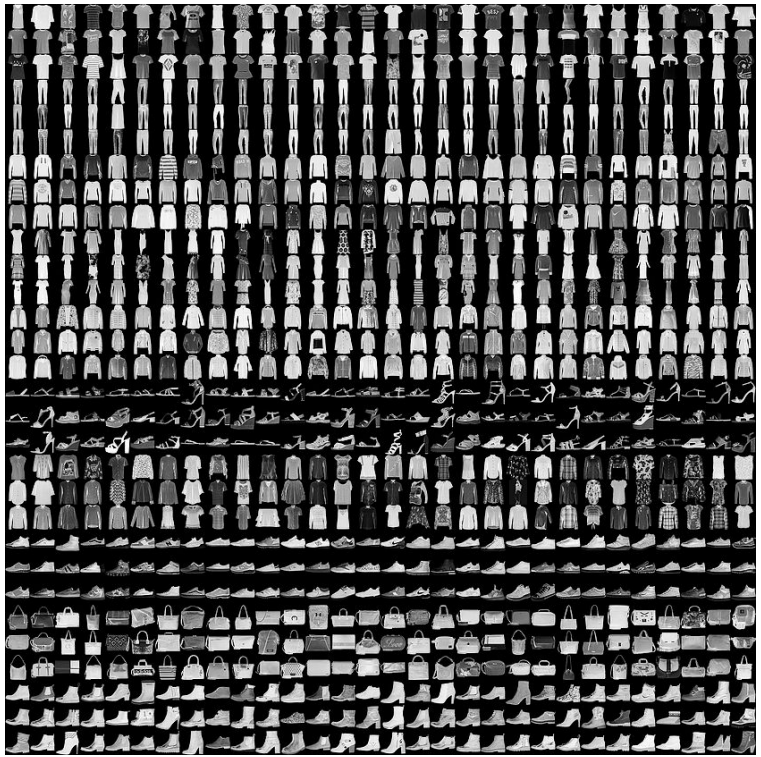
Quoted from their website:

- **MNIST is too easy**. Convolutional nets can achieve 99.7% on MNIST. Classic machine learning algorithms can also achieve 97% easily. Most pairs of MNIST digits can be distinguished pretty well by just one pixel.
- **MNIST is overused**. In this April 2017 Twitter thread, Google Brain research scientist and deep learning expert Ian Goodfellow calls for people to move away from MNIST.
- **MNIST can not represent modern CV tasks**, as noted in this April 2017 Twitter thread, deep learning expert/Keras author François Chollet.

Material: https://github.com/zalandoresearch/fashion-mnist

Introduction: Fashion-MNIST Dataset





Material: https://github.com/zalandoresearch/fashion-mnist

How to import?

- Loading data with Python (requires NumPy)
  - Use utils/mnist_reader from https://github.com/zalandoresearch/fashion-mnist

```
import mnist_reader
X_train, y_train = mnist_reader.load_mnist('data/fashion', kind='train')
X_test, y_test = mnist_reader.load_mnist('data/fashion', kind='t10k')
```

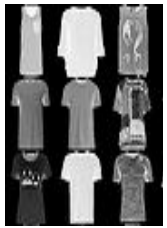
- Loading data with Tensorflow
  - Make sure you have downloaded the data and placed it in data/fashion. Otherwise, Tensorflow will download and use the original MNIST.

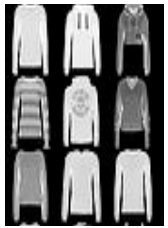
```
from tensorflow.examples.tutorials.mnist import input_data
data = input_data.read_data_sets('data/fashion')

data.train.next_batch(BATCH_SIZE)
```

Material: https://github.com/zalandoresearch/fashion-mnist

Feature Extraction

- We compared three different feature representation:
  - Raw pixel features
  - ScatNet features
  - Pretrained ResNet18 last-layer features



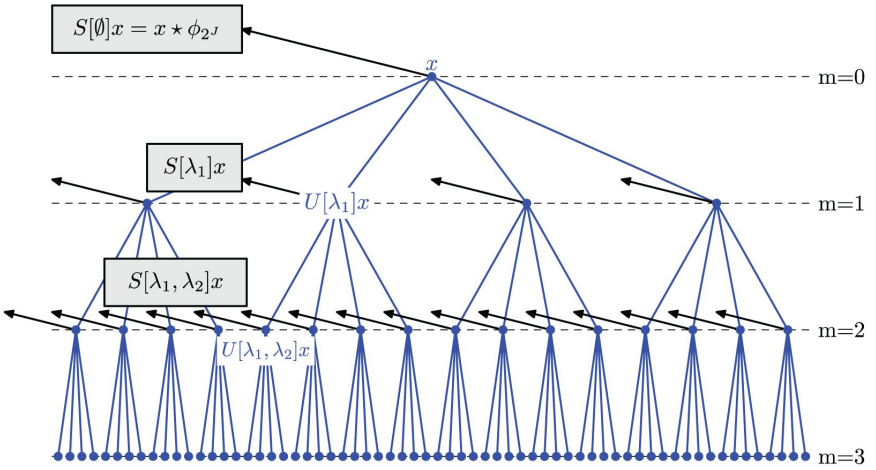




Feature Extraction(1): ScatNet

- The maximum scale of the transform: J=3
- The maximum scattering order: M=2
- The number of different orientations: L=1

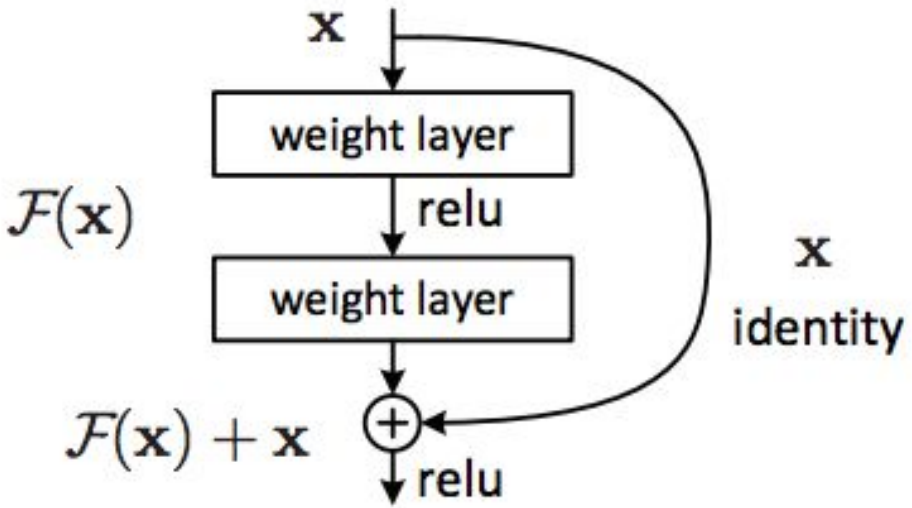
The dimension of the final features is 176



https://arxiv.org/pdf/1203.1513.pdf

Feature Extraction(2): ResNet

- Used pretrained 18 layers Residual Network from ImageNet
- We take the hidden representation right before the last fully-connected layer, which has the dimension of 512



https://arxiv.org/abs/1512.03385

Data Visualization

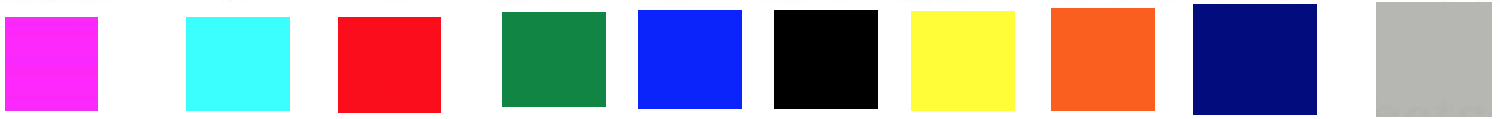
- Then, we visualized three different feature representation by the following 4 different dimension reduction methods:
  - Principal Component Analysis (PCA)
  - Locally Linear Embedding (LLE)
  - t-Distributed Stochastic Neighbor Embedding (t-SNE)
  - Uniform Manifold Approximation and Projection (UMAP)

index	0	1	2	3	4	5	6	7	8	9
Type	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
Color	'magenta'	'cyan'	'red'	'green'	'blue'	'black'	'yellow'	'orange'	'navy'	'gray'

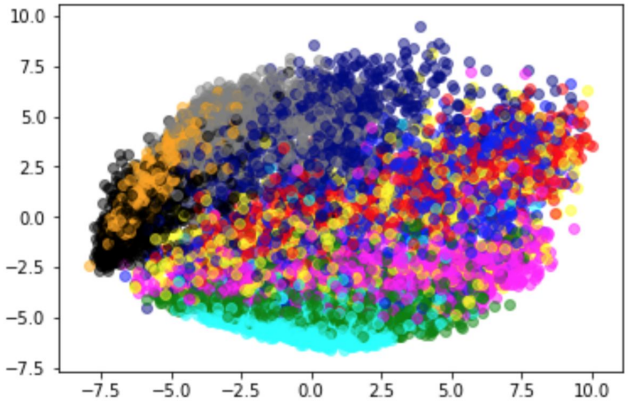
Data Visualization

- Then, we visualized three different feature representation by the following 4 different dimension reduction methods:
  - Principal Component Analysis (PCA)
  - Locally Linear Embedding (LLE)
  - t-Distributed Stochastic Neighbor Embedding (t-SNE)
  - Uniform Manifold Approximation and Projection (UMAP)

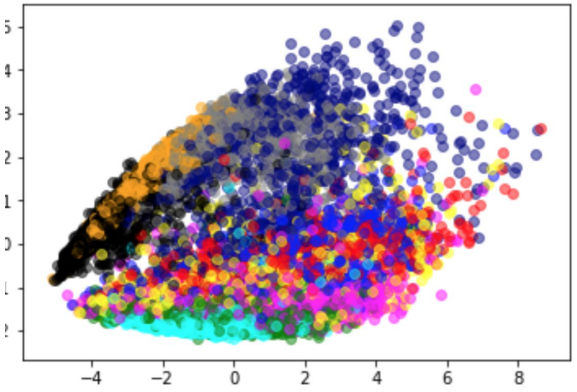
index	0	1	2	3	4	5	6	7	8	9
Type	T-shirt/top	Trouser	Pullover	Dress	Coat	Sandal	Shirt	Sneaker	Bag	Ankle boot
Color	'magenta'	'cyan'	'red'	'green'	'blue'	'black'	'yellow'	'orange'	'navy'	'gray'



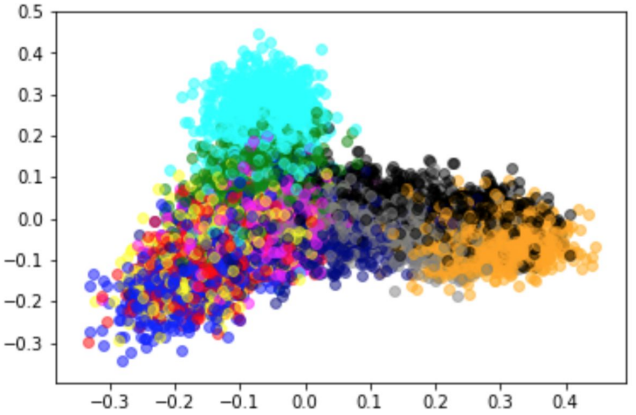
Data Visualization (1): PCA



Raw Features



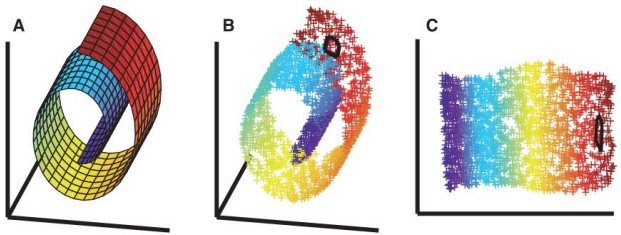
ScatNet Features



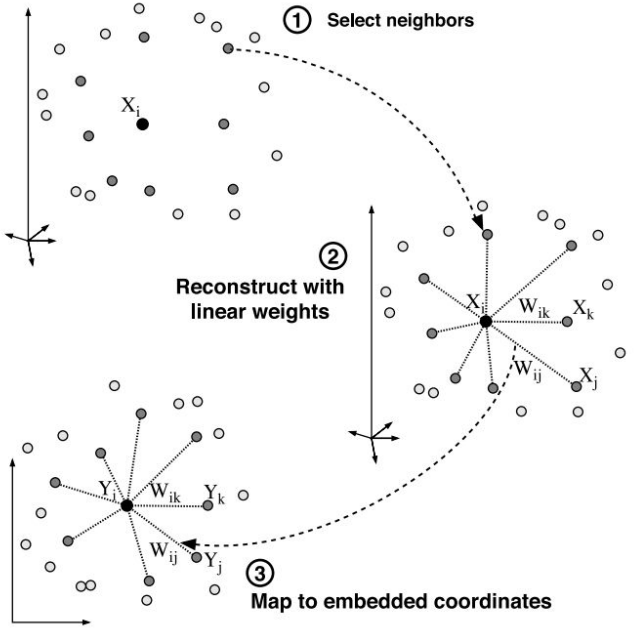
ResNet Features

- Normalization, Covariance Matrix, SVD, Project to top K eigen-vectors
- Linear dimension reduction methods:
  - not that obviously difference between labels

Data Visualization (2): LLE



1. Compute the neighbors of each data point, $\vec{X}_i$.
2. Compute the weights W_{ij} that best reconstruct each data point $\vec{X}_i$ from its neighbors, minimizing the cost in Equation (1) by constrained linear fits.
3. Compute the vectors $\vec{Y}_i$ best reconstructed by the weights W_{ij}, minimizing the quadratic form in Equation (2) by its bottom nonzero eigenvectors.



http://www.robots.ox.ac.uk/~az/lectures/ml/lle.pdf

Data Visualization (2): LLE

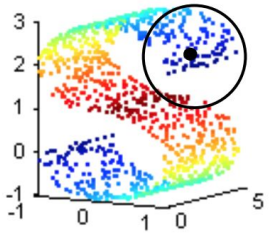
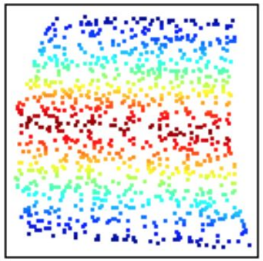
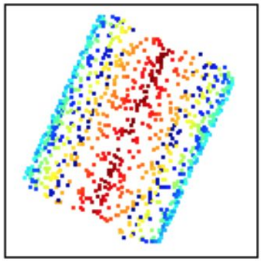


Figure 1. A nonlinear S-manifold consisting of 1000 data points



a) the result obtained by LLE



b) the result obtained by PCA

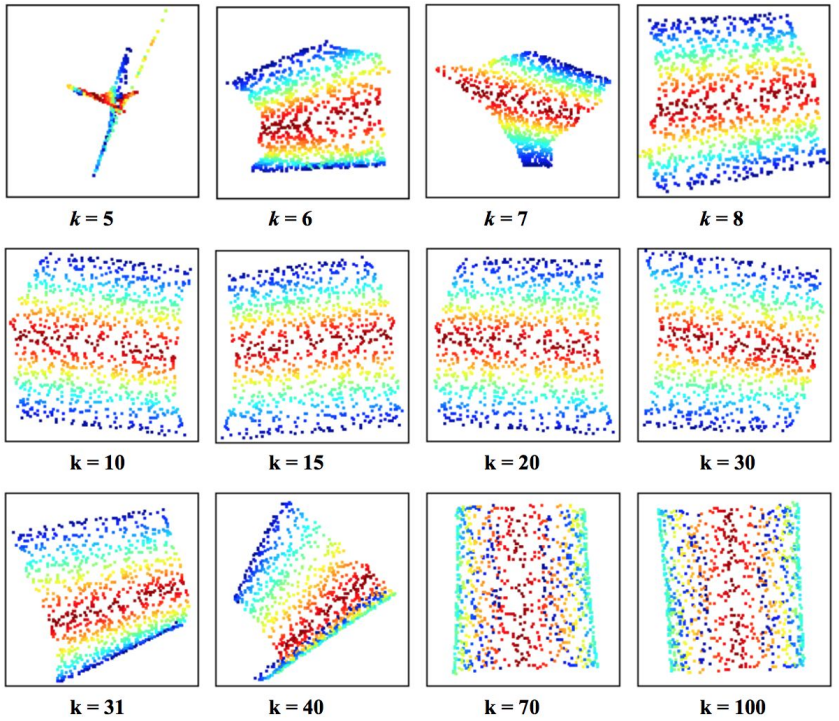
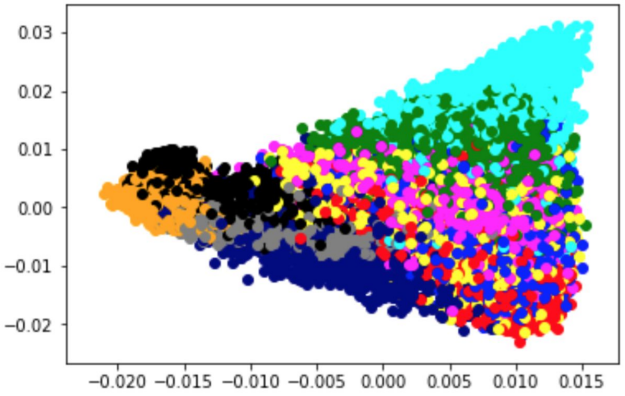


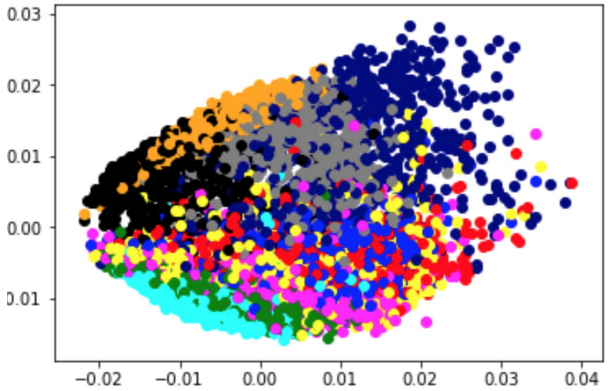
Figure 3. Embeddings of the 2-dimensional S-manifold, computed for different choices of the number of the nearest neighbours *k* by LLE

https://pdfs.semanticscholar.org/6adc/19cf4404b9f1224a1a027022e40ac77218f5.pdf

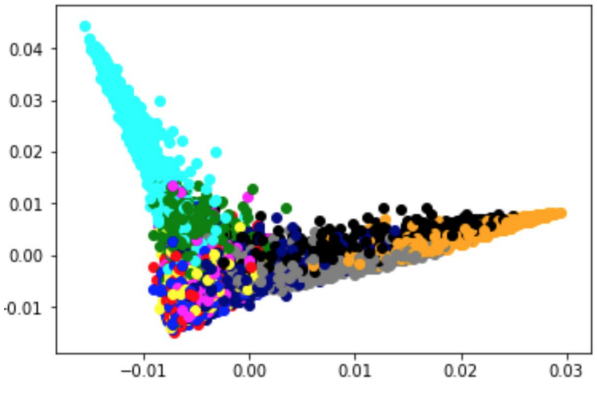
Data Visualization (2): LLE



Raw Features



ScatNet Features



ResNet Features

- Non-linear dimension reduction that is good at capture "streamline" structure

Data Visualization (3): t-SNE

- Use Gaussian pdf to approximate the high dimension distribution
- Use t distribution for low dimension distribution
- Use KL Divergence as cost function for gradient descent

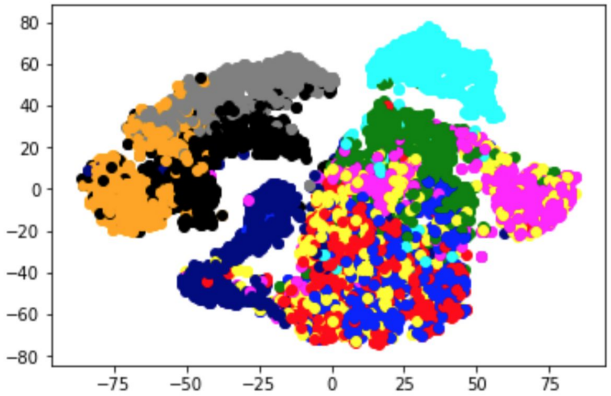
$$p_{ij} = \frac{\exp \frac{\left(-\|x_i - x_j\|^2\right)}{2\sigma^2}}{\sum \frac{\exp(-\|x_k - x_l\|^2)}{2\sigma^2}} \qquad q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\Sigma \left(1 + \|y_k - y_l\|^2\right)^{-1}}$$

$$C = KL(P||Q) = \Sigma_i \Sigma_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

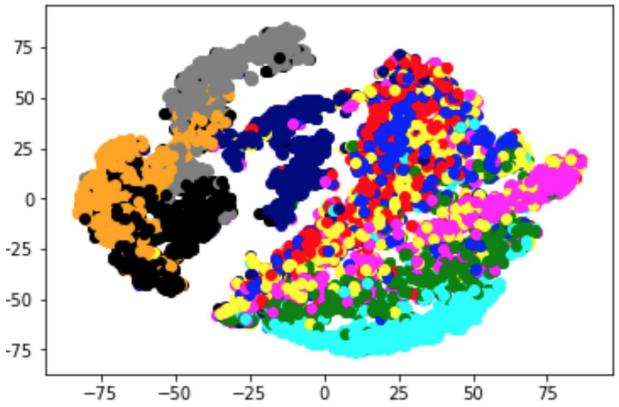
$$\frac{\delta C}{\delta y_i} = 4\Sigma_j \left(p_{ij} - q_{ij}\right)\left(y_i - y_j\right)\left(1 + \|y_i - y_j\|^2\right)^{-1}$$

http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf

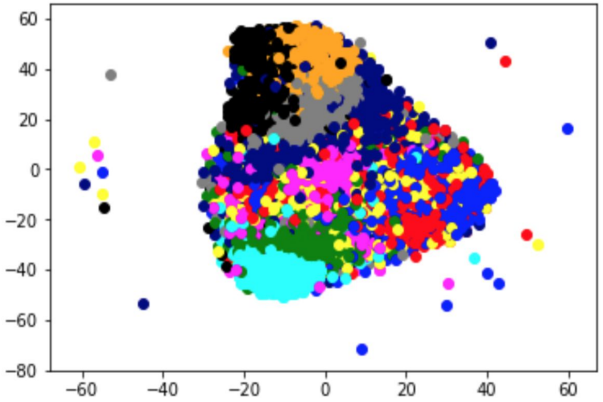
Data Visualization (3): t-SNE



Raw Features



ScatNet Features



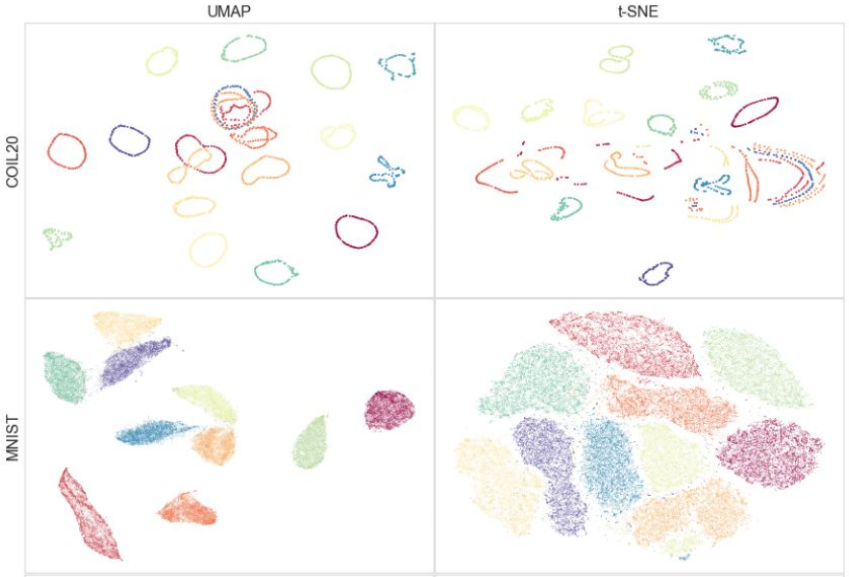
ResNet Features

- Block-like visualization due to the gaussian approximation

Data Visualization (4): UMAP

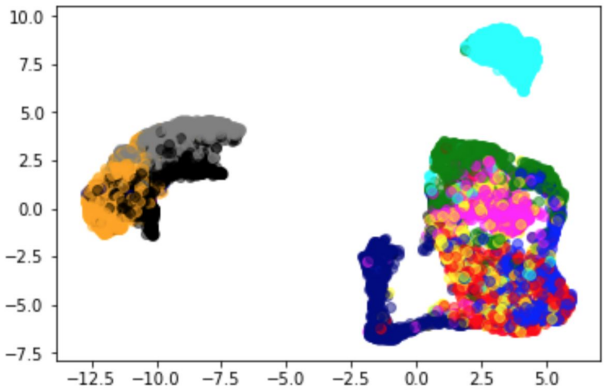
- The algorithm is founded on three assumptions about the data
  - The Riemannian metric is locally constant (or can be approximated);
  - The data is uniformly distributed on Riemannian manifold;
  - The manifold is locally connected.

https://arxiv.org/pdf/1802.03426.pdf

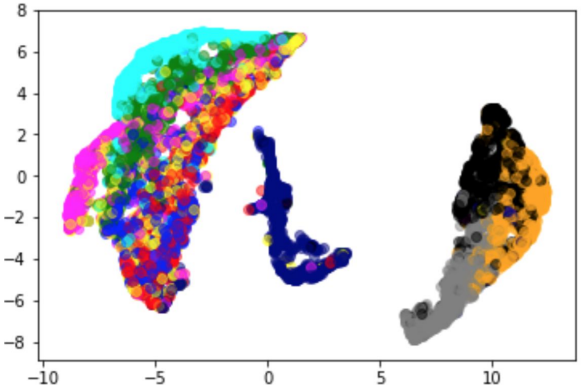


Data Visualization (4): UMAP

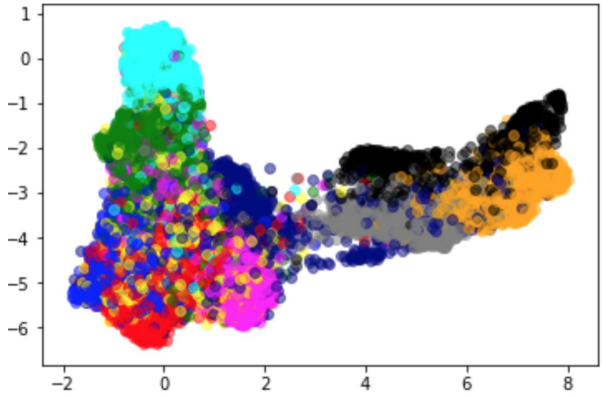
https://github.com/lmcinnes/umap



Raw Features



ScatNet Features



ResNet Features

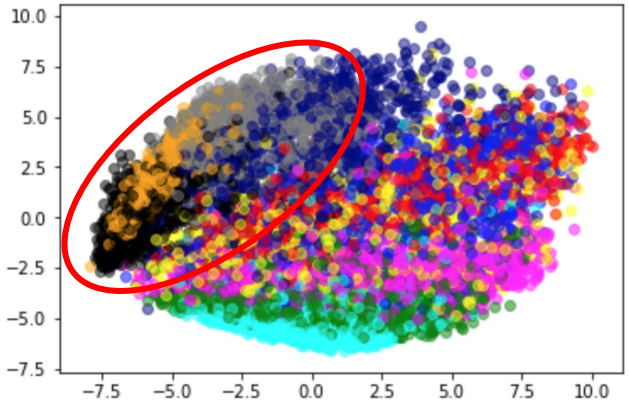
- Much more Faster in training process, which implies it can handle large datasets and high dimensional data

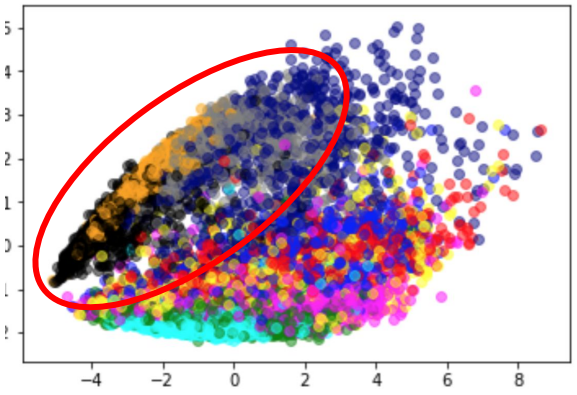
Any News from Visualization?

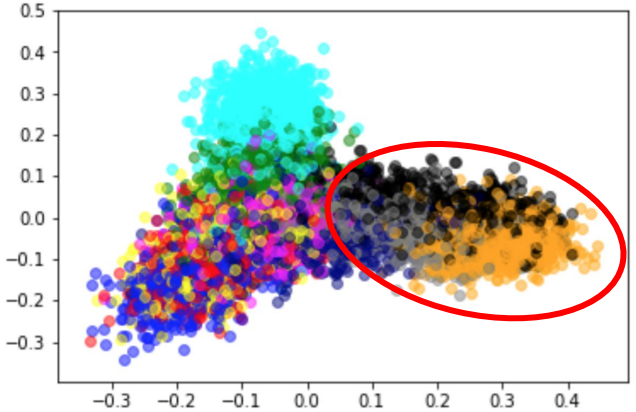
- Is there different patterns between different visualization methods?
- Is there clear separation of different classes?
- Is there any groups that tend to cluster together?

- Let's look closer!

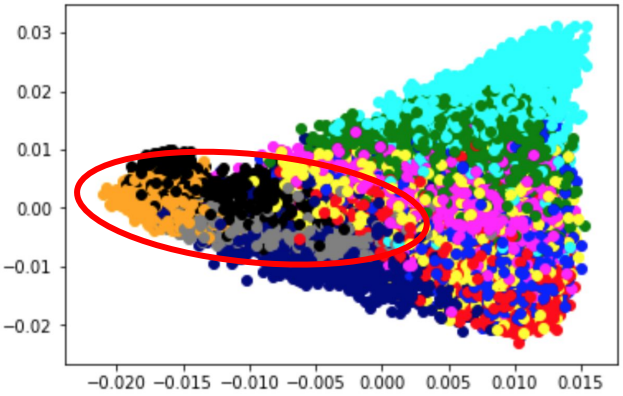
PCA

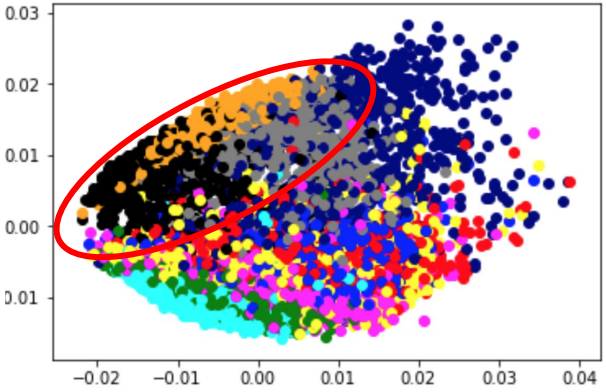

10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0

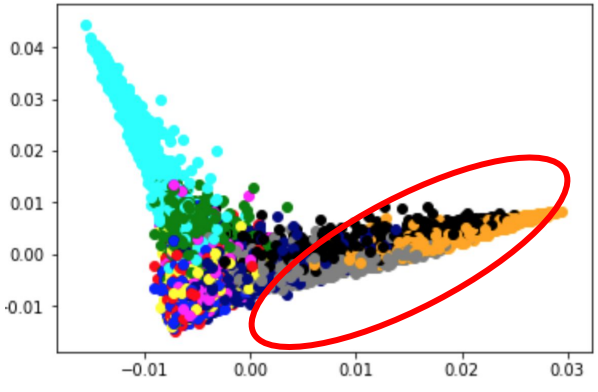

−4 −2 0 2 4 6 8


0.5
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.3 −0.2 −0.1 0.0 0.1 0.2 0.3 0.4

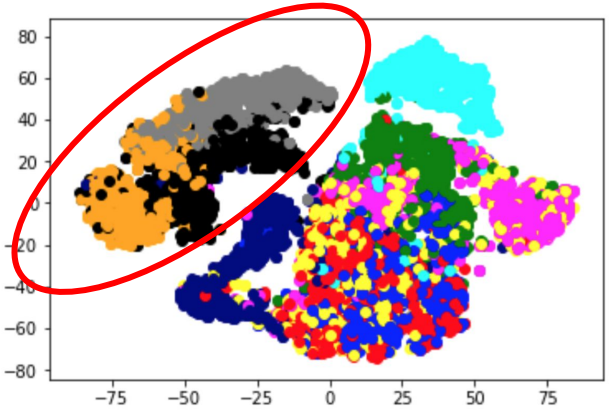
LLE

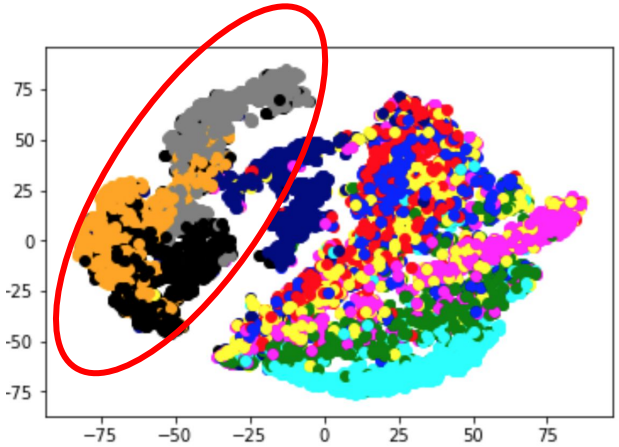

0.03
0.02
0.01
0.00
−0.01
−0.02
−0.020 −0.015 −0.010 −0.005 0.000 0.005 0.010 0.015

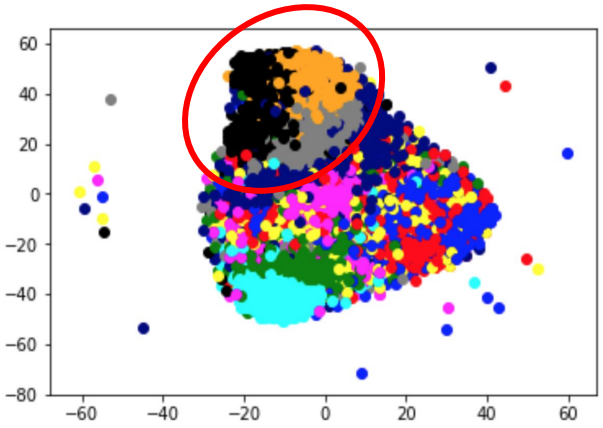

0.03
0.02
0.01
0.00
0.01
−0.02 −0.01 0.00 0.01 0.02 0.03 0.04


0.04
0.03
0.02
0.01
0.00
−0.01
−0.01 0.00 0.01 0.02 0.03

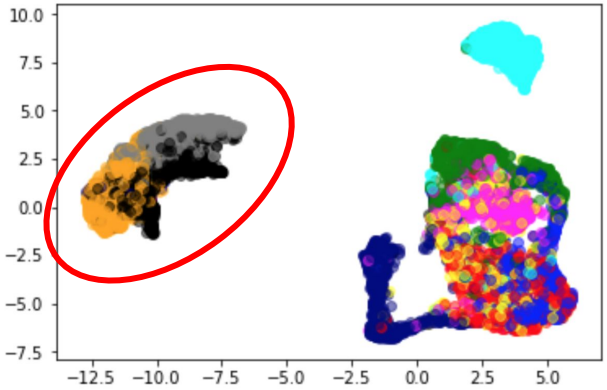
t-SNE

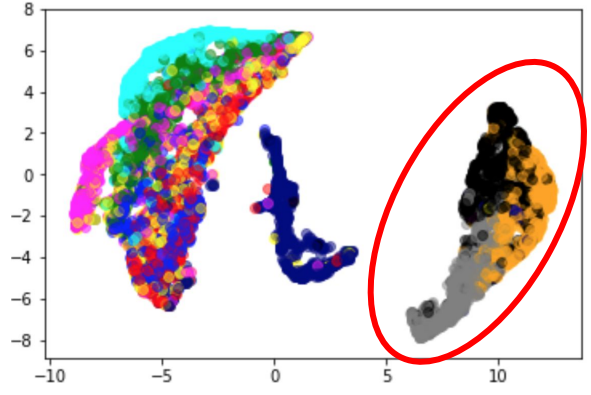

80
60
40
20
0
−20
−40
−60
−80
−75
−50
−25
0
25
50
75

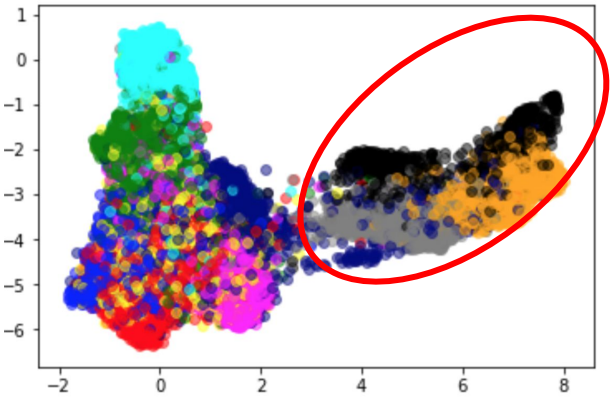

75
50
25
0
−25
−50
−75
−75
−50
−25
0
25
50
75


60
40
20
0
−20
−40
−60
−80
−60
−40
−20
0
20
40
60

UMAP

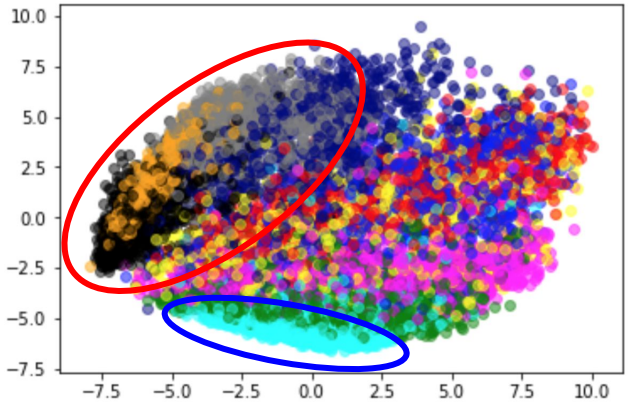

10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−12.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0

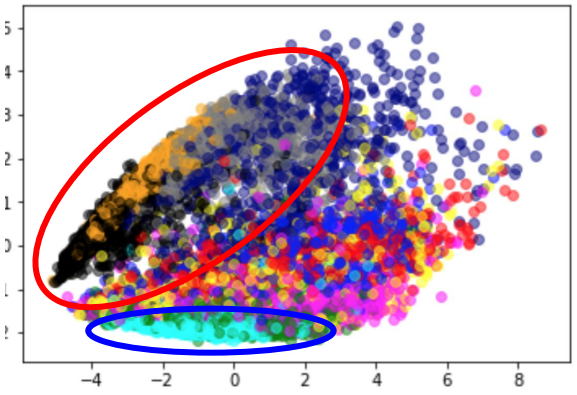

8
6
4
2
0
−2
−4
−6
−8
−10
−5
0
5
10

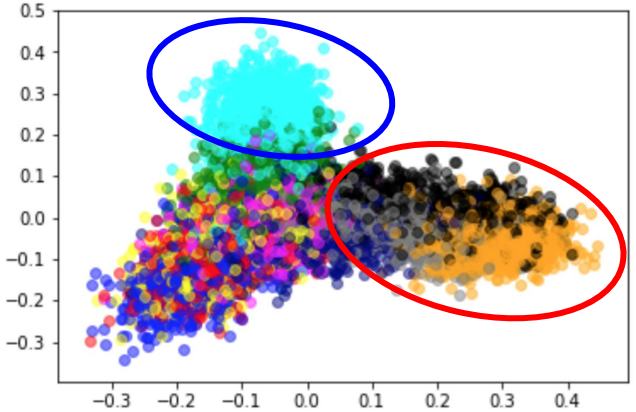

1
0
−1
−2
−3
−4
−5
−6
−2
0
2
4
6
8

Sneaker,
Sandal,
Ankle boot

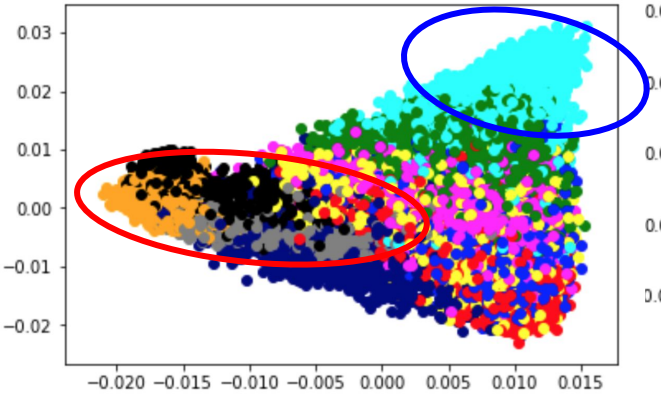
PCA

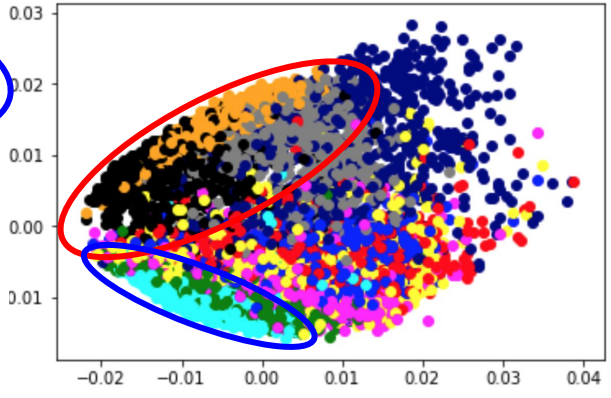

10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0

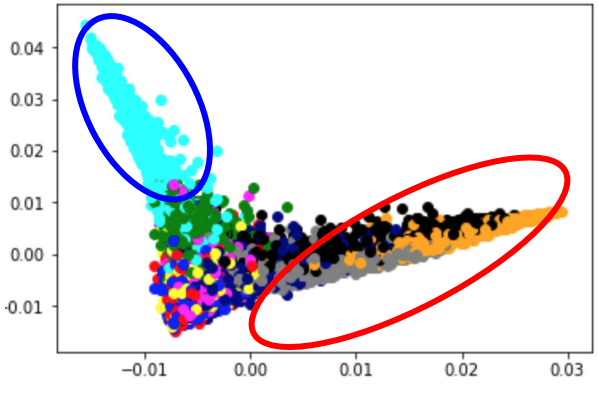

−4 −2 0 2 4 6 8


0.5
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.3 −0.2 −0.1 0.0 0.1 0.2 0.3 0.4

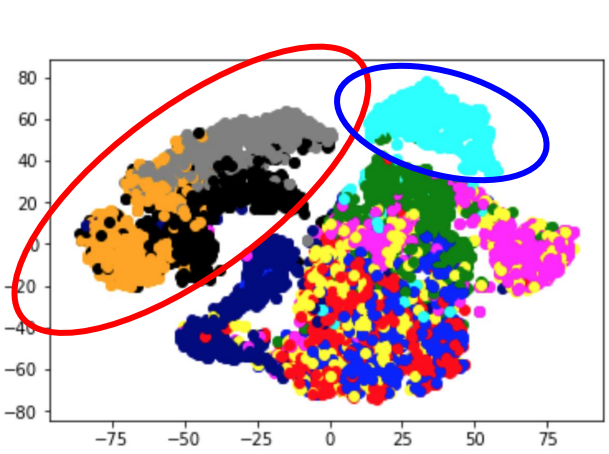
LLE

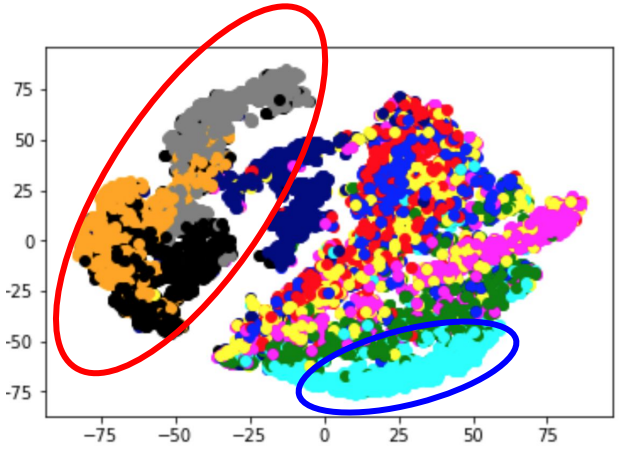

0.03
0.02
0.01
0.00
−0.01
−0.02
−0.020 −0.015 −0.010 −0.005 0.000 0.005 0.010 0.015

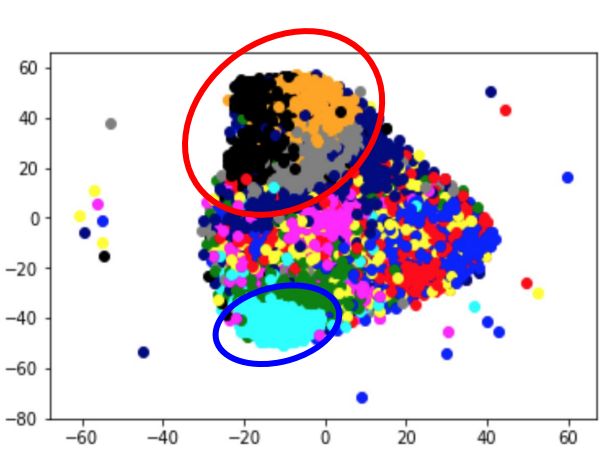

0.03
0.02
0.01
0.00
0.01
−0.02 −0.01 0.00 0.01 0.02 0.03 0.04


0.04
0.03
0.02
0.01
0.00
−0.01
−0.01 0.00 0.01 0.02 0.03

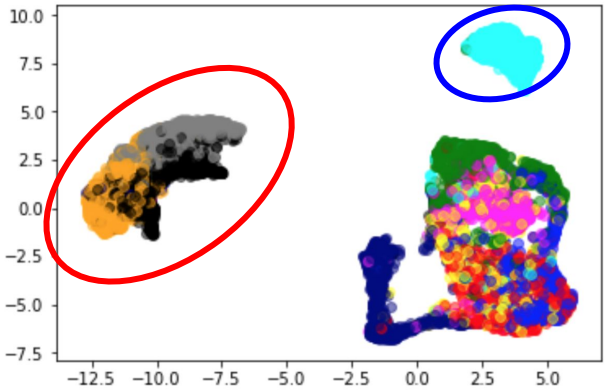
t-SNE

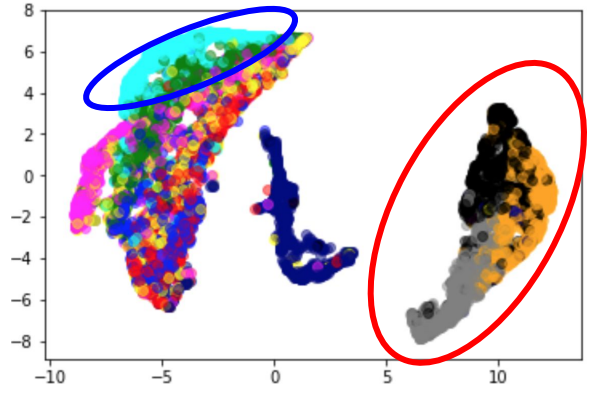

80
60
40
20
0
−20
−40
−60
−80
−75
−50
−25
0
25
50
75

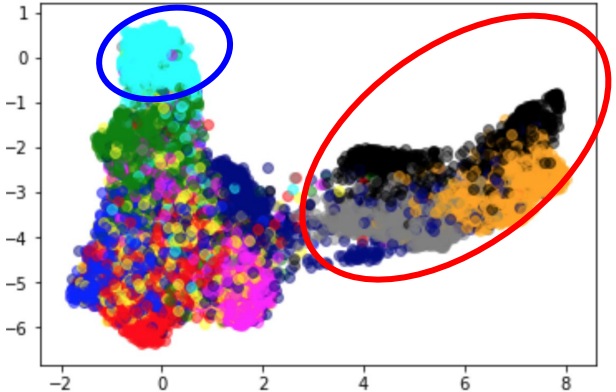

75
50
25
0
−25
−50
−75
−75
−50
−25
0
25
50
75


60
40
20
0
−20
−40
−60
−80
−60
−40
−20
0
20
40
60

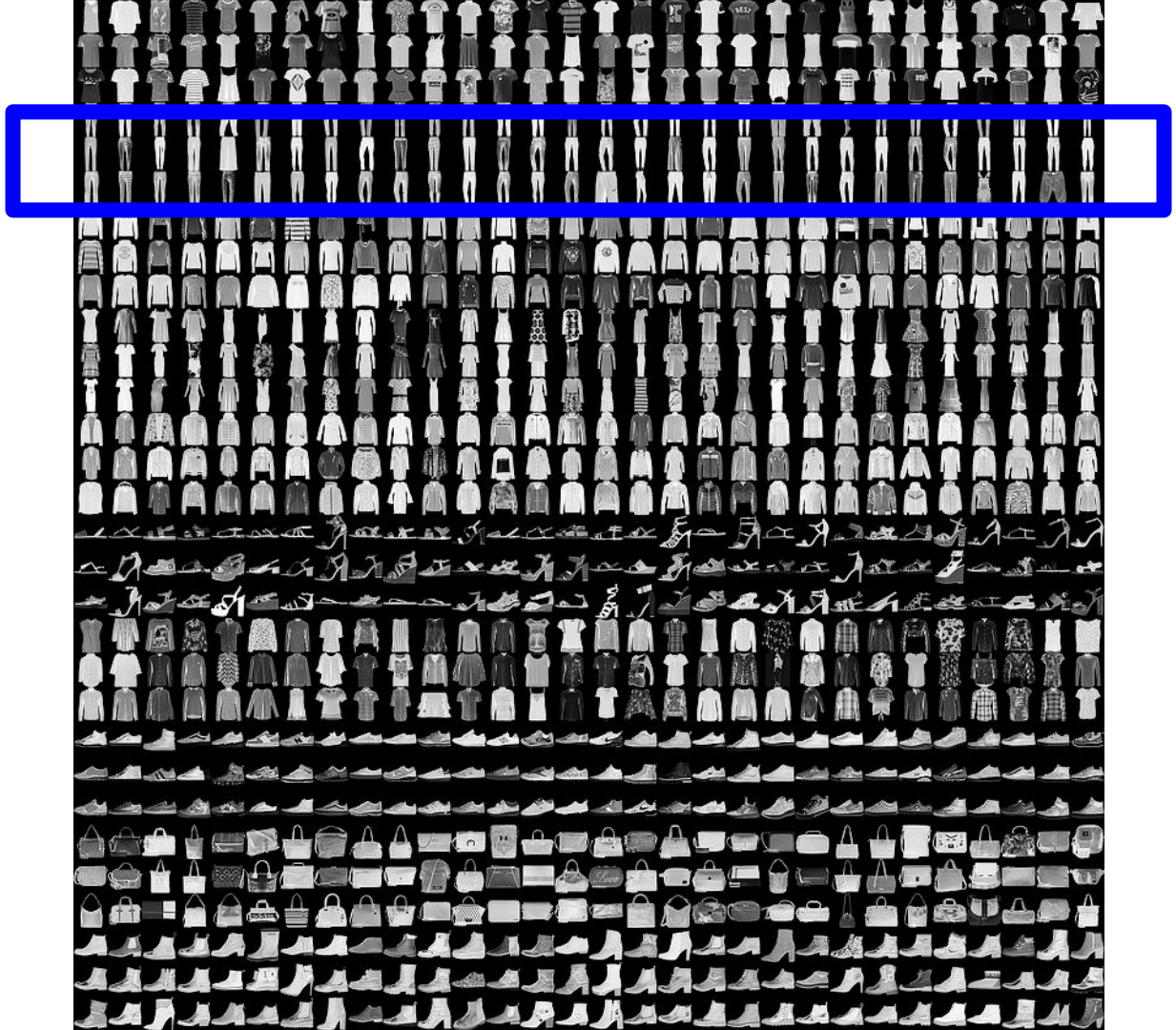
UMAP


10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−12.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0

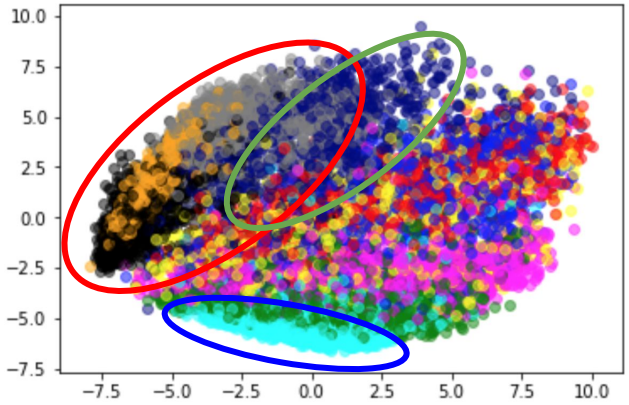

8
6
4
2
0
−2
−4
−6
−8
−10
−5
0
5
10

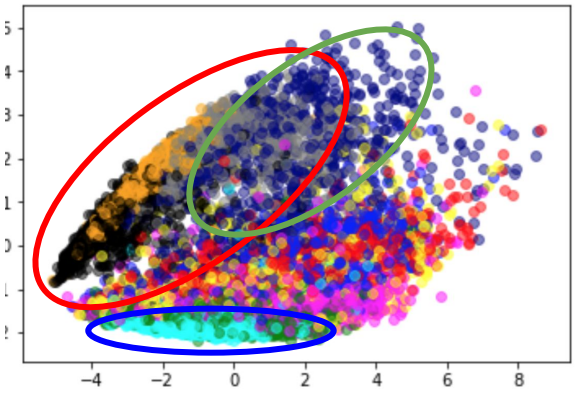

1
0
−1
−2
−3
−4
−5
−6
−2
0
2
4
6
8

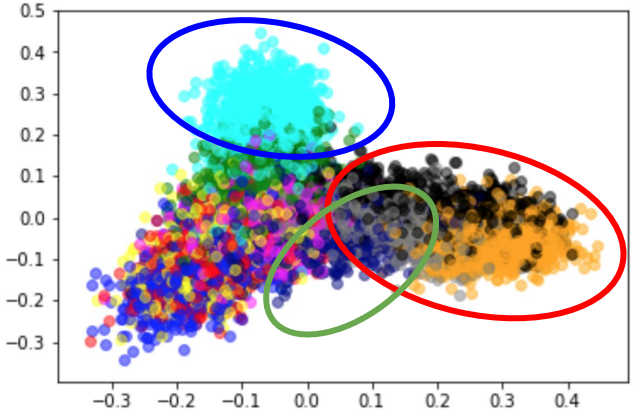
Trouser



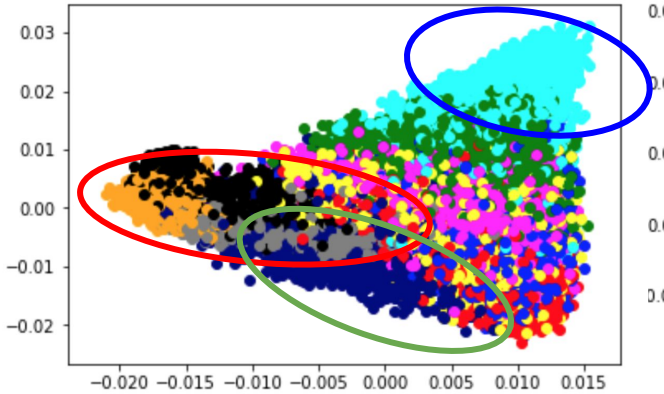
PCA

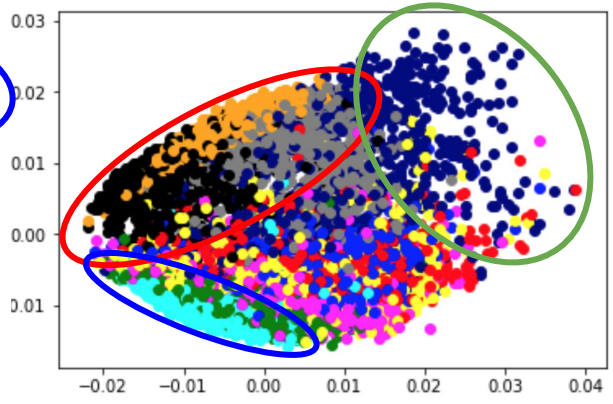

10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0

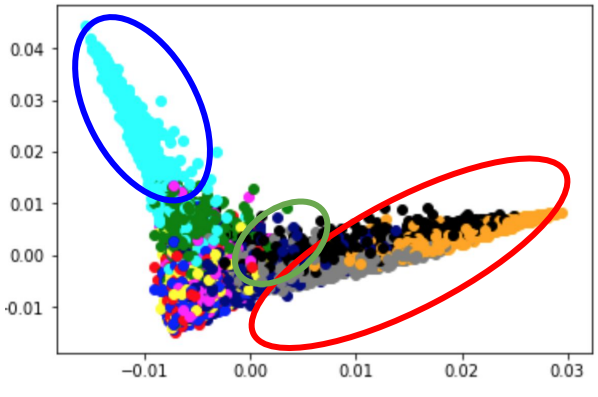

−4 −2 0 2 4 6 8


0.5
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.3 −0.2 −0.1 0.0 0.1 0.2 0.3 0.4

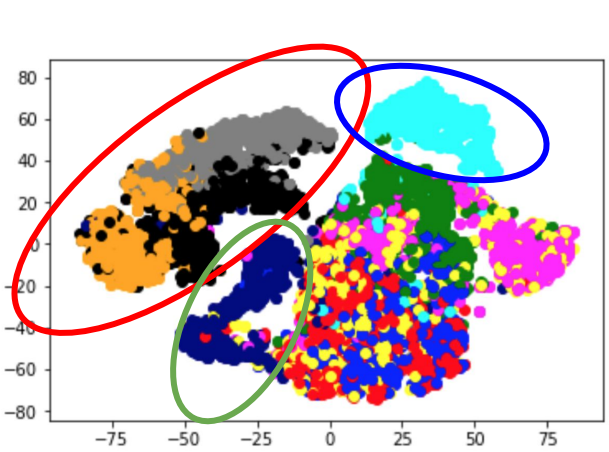
LLE

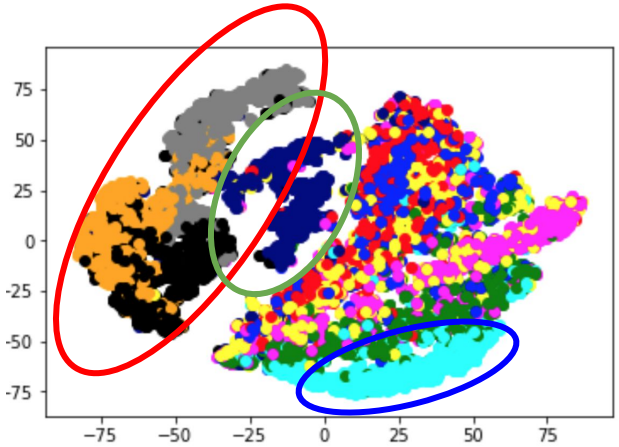

0.03
0.02
0.01
0.00
−0.01
−0.02
−0.020 −0.015 −0.010 −0.005 0.000 0.005 0.010 0.015

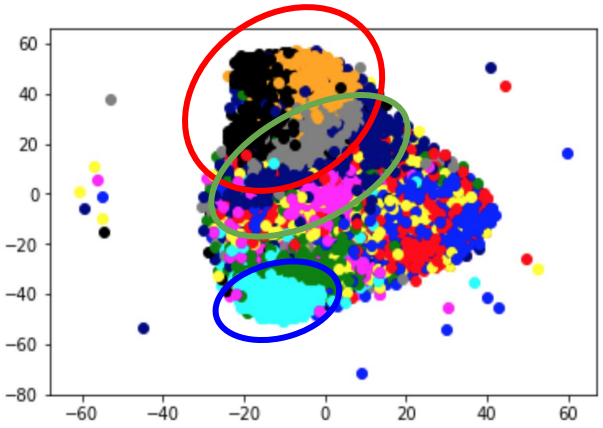

0.03
0.02
0.01
0.00
0.01
−0.02 −0.01 0.00 0.01 0.02 0.03 0.04


0.04
0.03
0.02
0.01
0.00
−0.01
−0.01 0.00 0.01 0.02 0.03

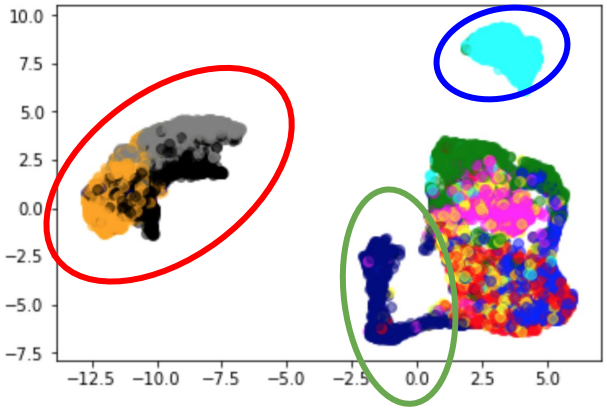
t-SNE

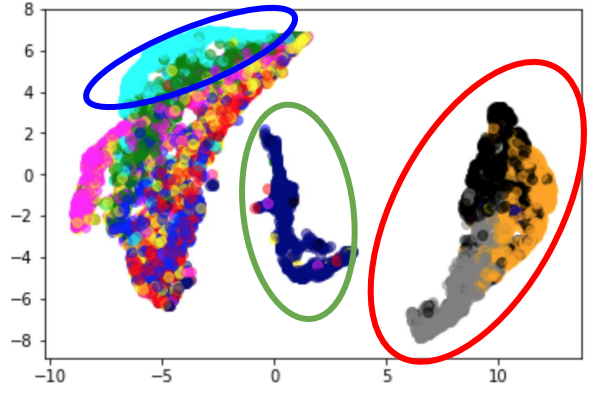

80
60
40
20
0
−20
−40
−60
−80
−75
−50
−25
0
25
50
75

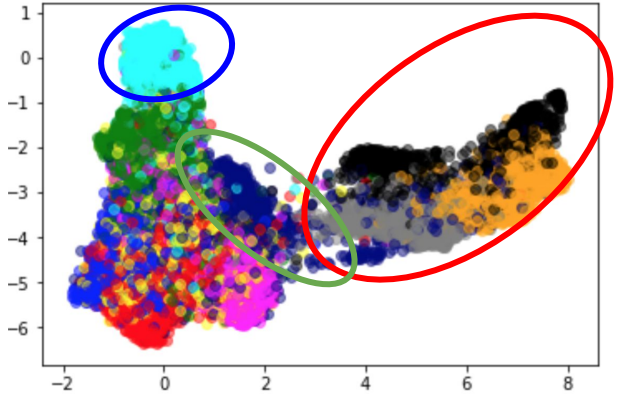

75
50
25
0
−25
−50
−75
−75
−50
−25
0
25
50
75


60
40
20
0
−20
−40
−60
−80
−60
−40
−20
0
20
40
60

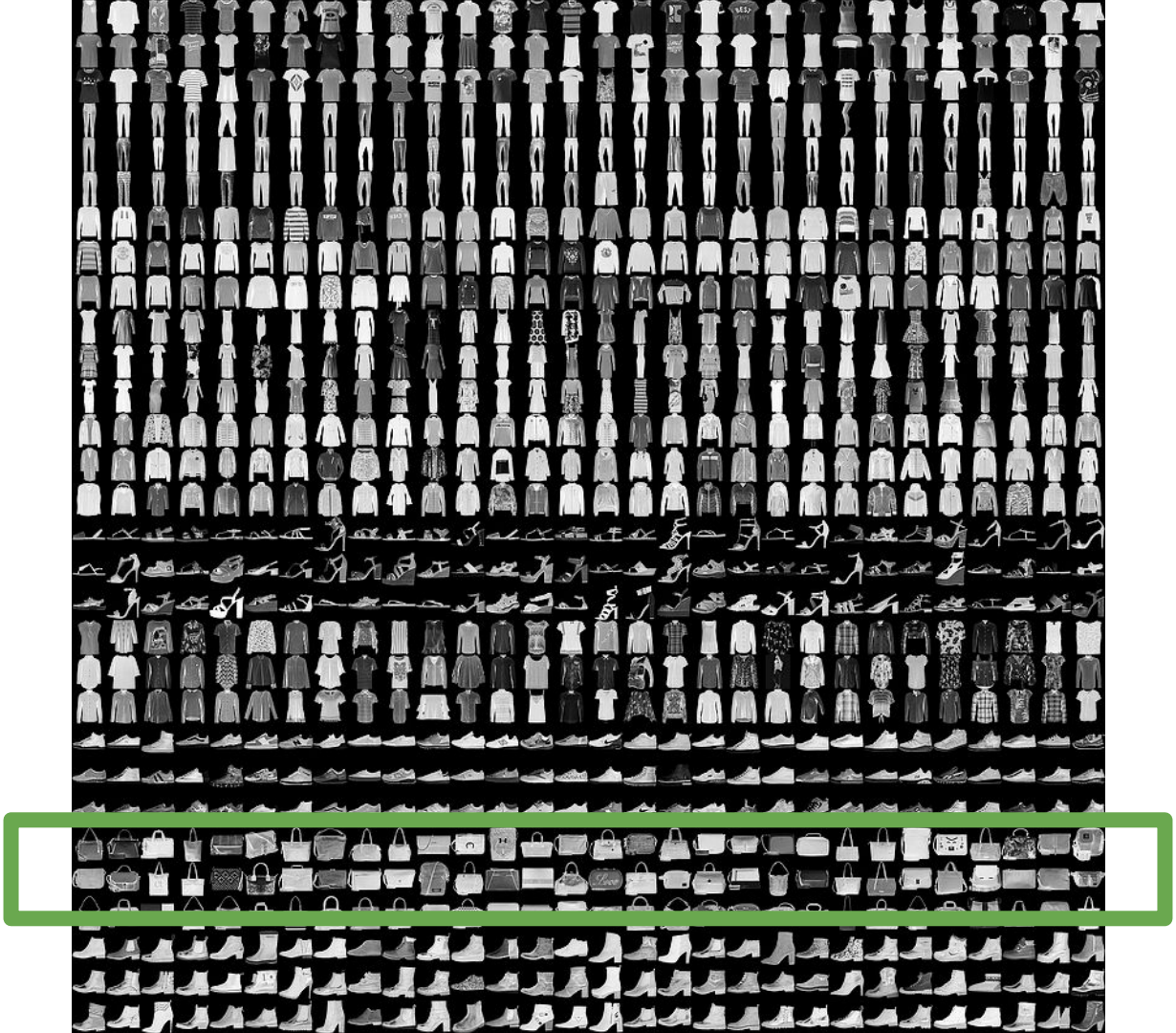
UMAP


10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−12.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0

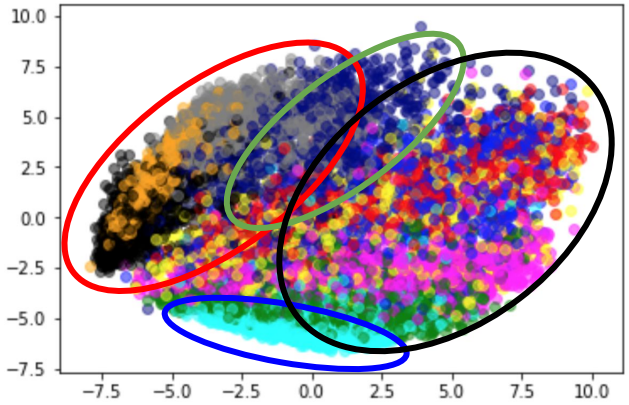

8
6
4
2
0
−2
−4
−6
−8
−10
−5
0
5
10

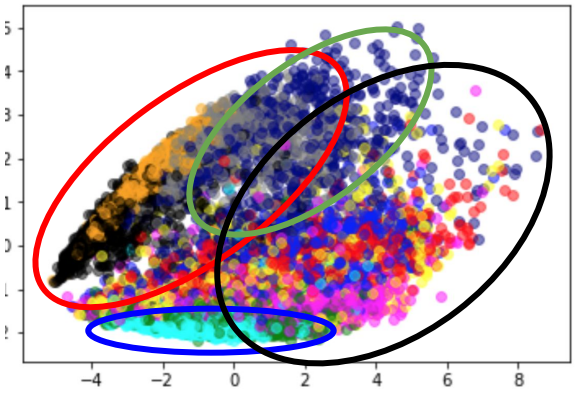

1
0
−1
−2
−3
−4
−5
−6
−2
0
2
4
6
8

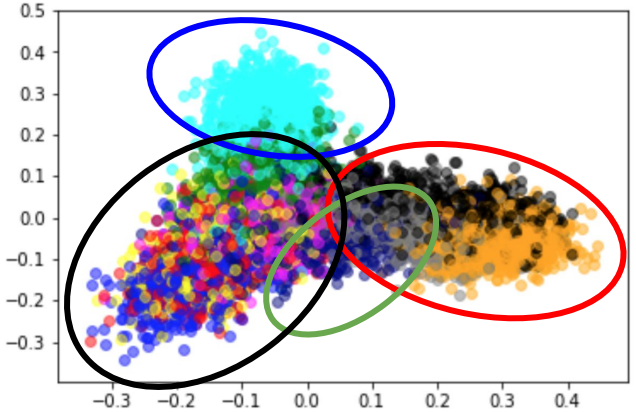
Bag



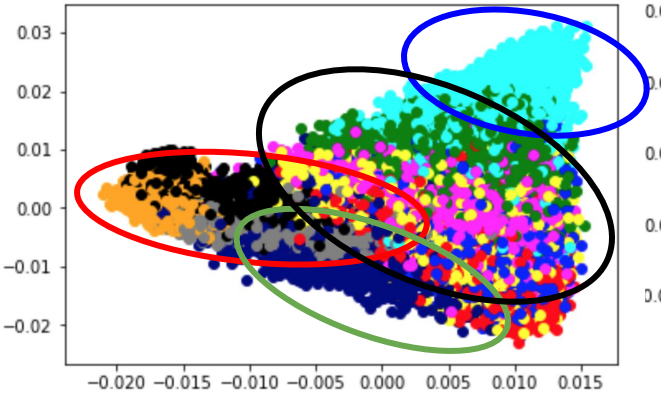
PCA

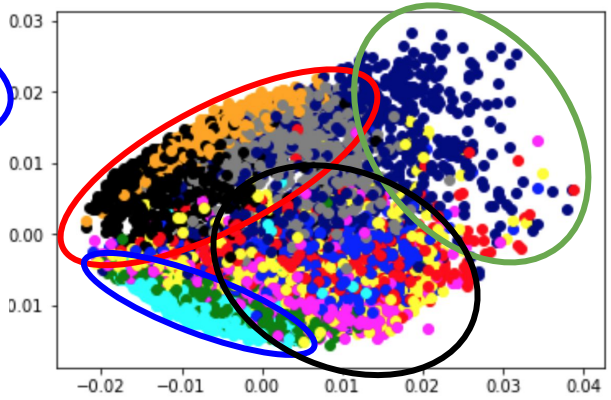

10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−7.5 −5.0 −2.5 0.0 2.5 5.0 7.5 10.0

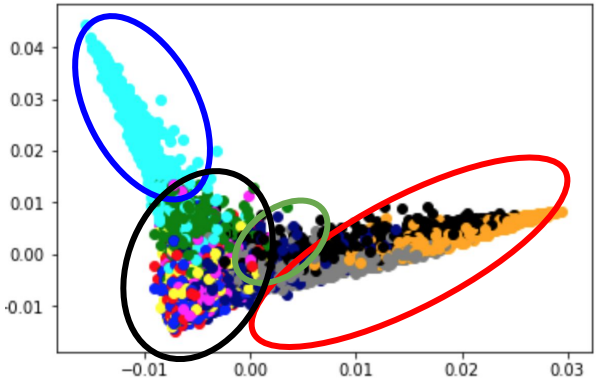

−4 −2 0 2 4 6 8


0.5
0.4
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
−0.3 −0.2 −0.1 0.0 0.1 0.2 0.3 0.4

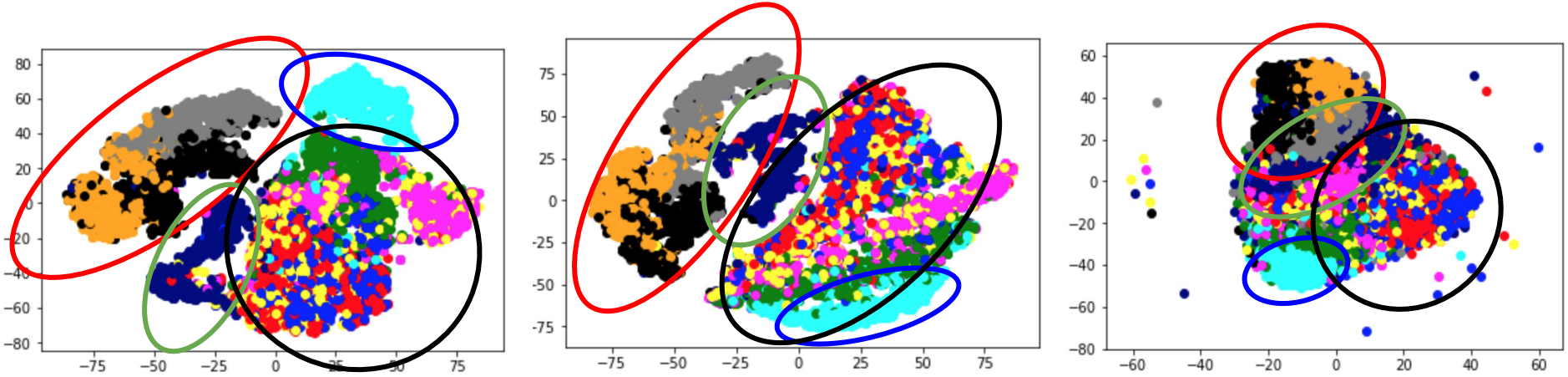
LLE


0.03
0.02
0.01
0.00
−0.01
−0.02
−0.020 −0.015 −0.010 −0.005 0.000 0.005 0.010 0.015

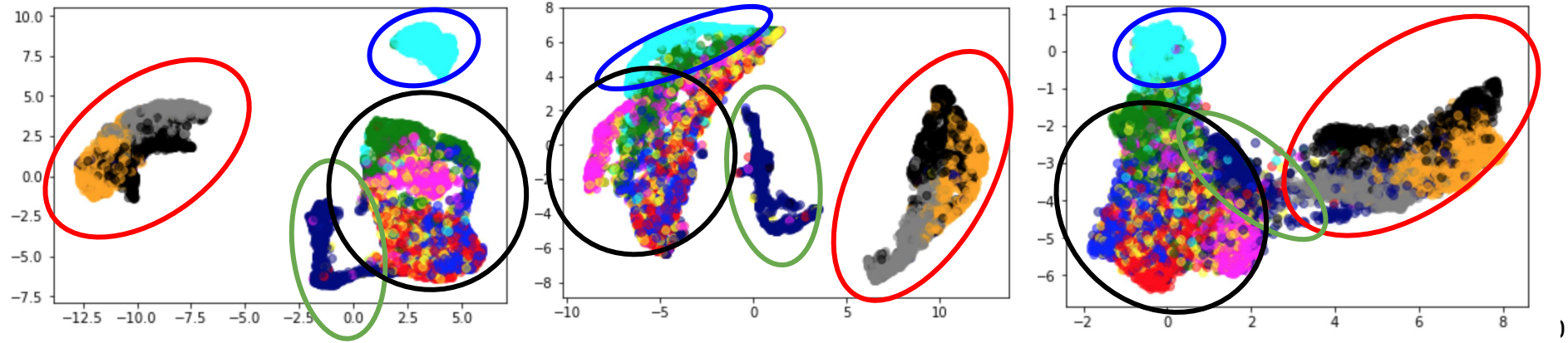

0.03
0.02
0.01
0.00
0.01
−0.02 −0.01 0.00 0.01 0.02 0.03 0.04


0.04
0.03
0.02
0.01
0.00
−0.01
−0.01 0.00 0.01 0.02 0.03

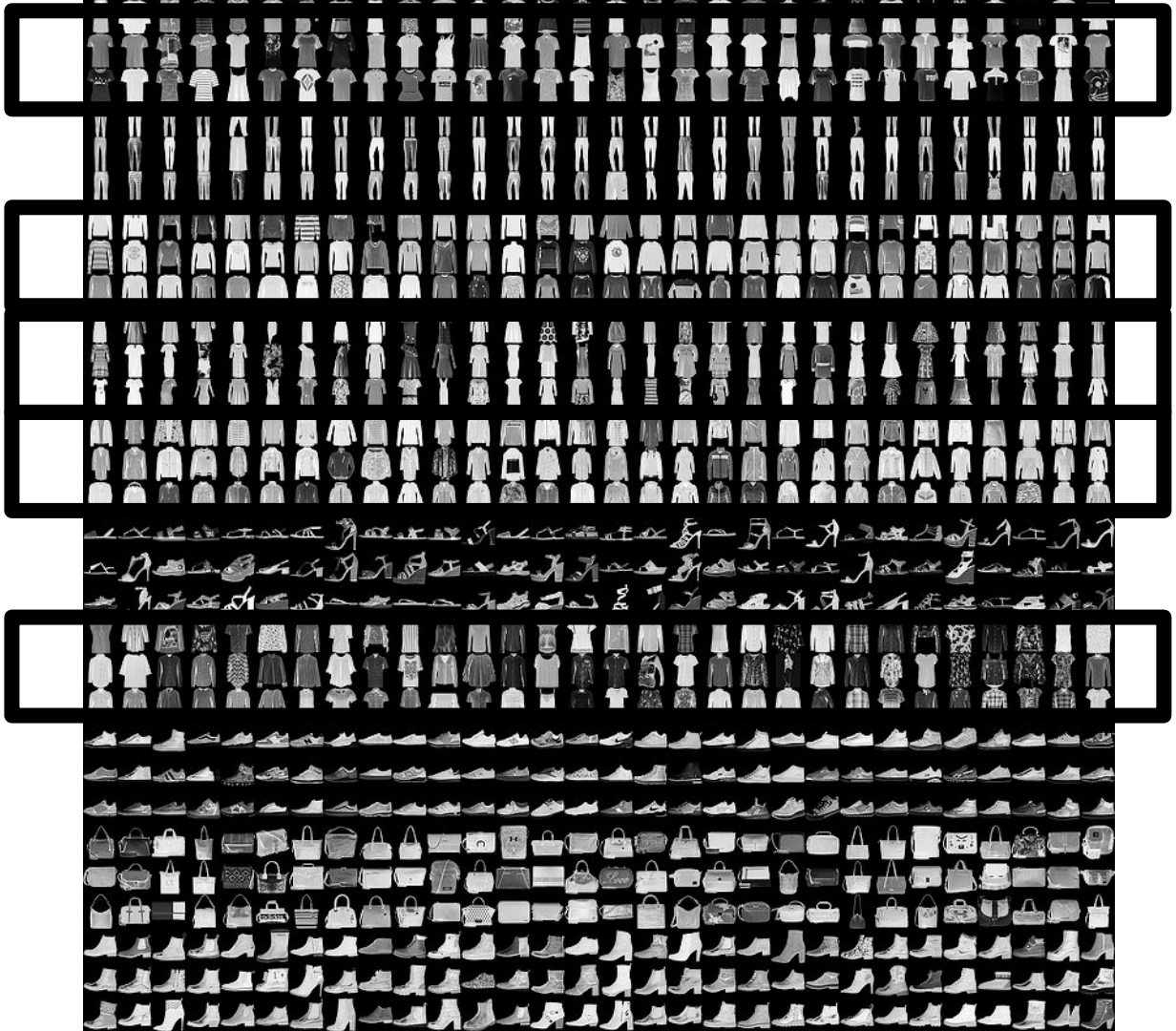
t-SNE



UMAP



T-Shirt,
Pullover,
Dress,
Coat,
Shirt

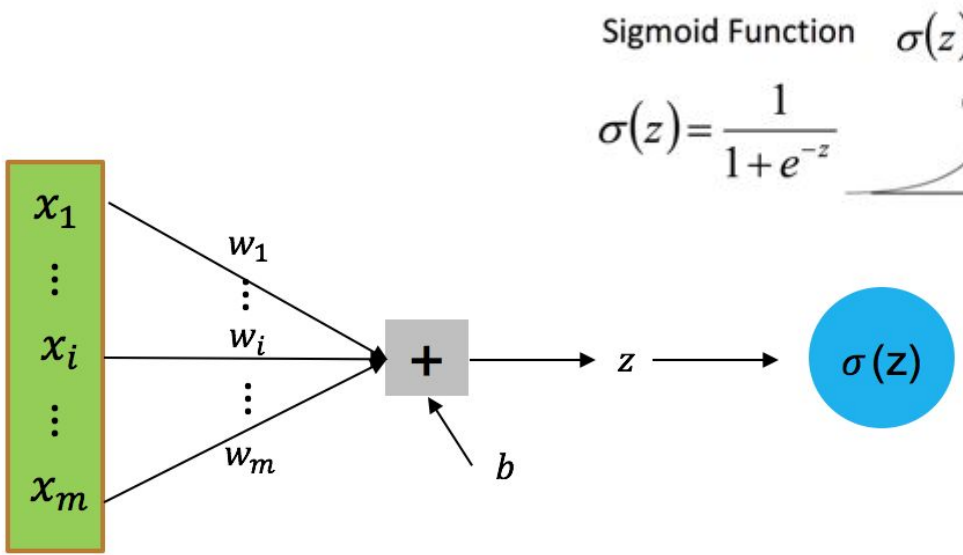


Simple Classification Models

- Logistic Regression
- Linear Discriminant Analysis
- Support Vector Machine
- Random Forest
- ...

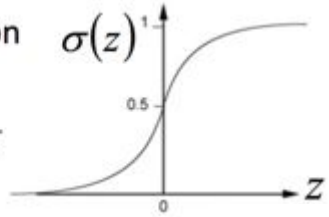
Simple Classification Models

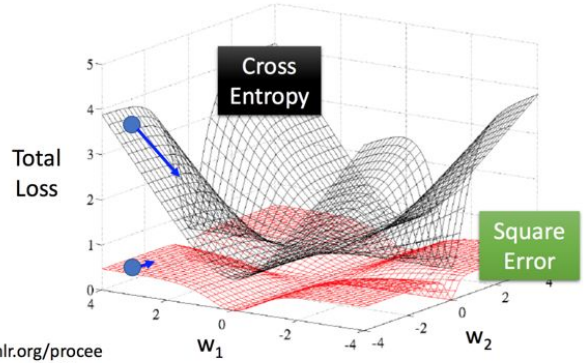
- Logistic Regression



Sigmoid Function

$$\sigma(z) = \frac{1}{1+e^{-z}}$$



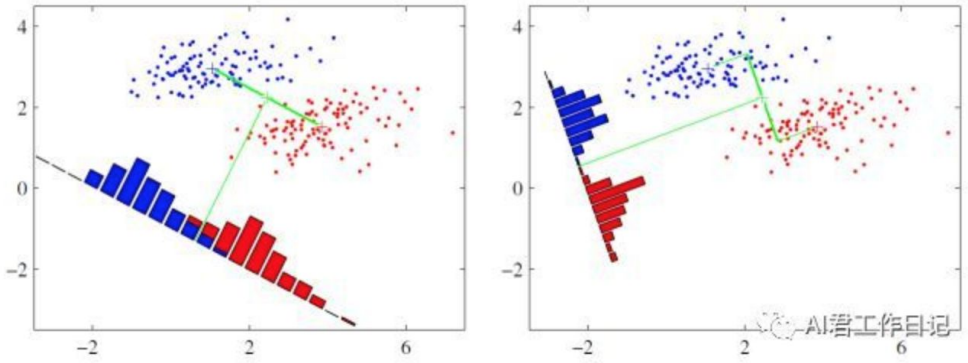


http://jmlr.org/proceedings/papers/v9/glorot10a/glorot10a.pdf

Simple Classification Models

- Linear Discriminant Analysis
  - maximize between class covariance
  - minimize within class covariance

$$S = \frac{\sigma^2_{\text{between}}}{\sigma^2_{\text{within}}} = \frac{(\vec{w} \cdot \vec{\mu}_1 - \vec{w} \cdot \vec{\mu}_0)^2}{\vec{w}^T \Sigma_1 \vec{w} + \vec{w}^T \Sigma_0 \vec{w}} = \frac{(\vec{w} \cdot (\vec{\mu}_1 - \vec{\mu}_0))^2}{\vec{w}^T (\Sigma_0 + \Sigma_1) \vec{w}}$$



Simple Classification Models

- Linear Support Vector Machine
  - Hard-margin

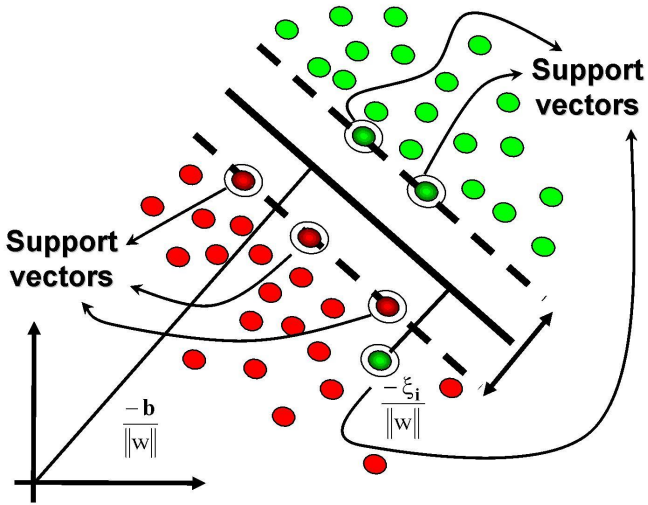
$$\vec{w} \cdot \vec{x}_i - b \geq 1, \text{ if } y_i = 1$$

or

$$\vec{w} \cdot \vec{x}_i - b \leq -1, \text{ if } y_i = -1.$$

  - Soft-margin

$$\left[\frac{1}{n}\sum_{i=1}^{n} \max\left(0, 1 - y_i(\vec{w} \cdot \vec{x}_i - b)\right)\right] + \lambda\|\vec{w}\|^2,$$

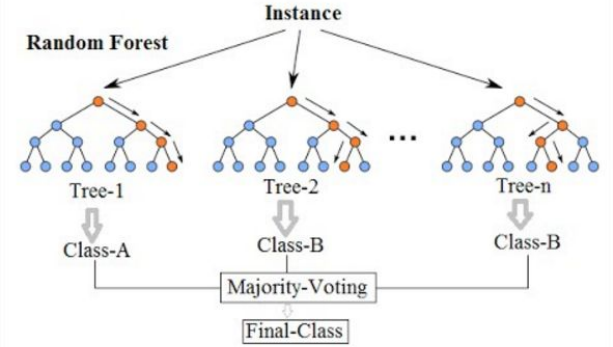


Simple Classification Models

- Random Forest
- An ensemble learning method that construct multiple decision trees
- Bagging (Bootstrap aggregating)

For $b = 1, \ldots, B$:

1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.
2. Train a classification or regression tree f_b on X_b, Y_b.



Simple Classification Results

Models	Raw Features 784	ScatNet Features 176	ResNet Features 512
Logistic Regression	84.60%	80.83%	79.03%
Linear Discriminant Analysis (LDA)	81.51%	79.96%	77.08%
Support Vector Machine (SVM)	78.84%	82.60%	73.63%
Random Forest	87.12%	83.06%	59.20%
Average	83.01%	81.61%	72.24%

Simple Classification Results

Models	Raw Features 784	ScatNet Features 176	ResNet Features 512	Average
Logistic Regression	84.60%	80.83%	79.03%	81.49%
Linear Discriminant Analysis (LDA)	81.51%	79.96%	77.08%	79.52%
Support Vector Machine (SVM)	78.84%	82.60%	73.63%	78.36%
Random Forest	87.12%	83.06%	59.20%	76.46%

Simple Classification Results

Models	Raw Features 784	ScatNet Features 176	ResNet Features 512
Logistic Regression	84.60%	80.83%	79.03%
Linear Discriminant Analysis (LDA)	81.51%	79.96%	77.08%
Support Vector Machine (SVM)	78.84%	82.60%	73.63%
Random Forest	87.12%	83.06%	59.20%

http://fashion-mnist.s3-website.eu-central-1.amazonaws.com

Fine-Tuning the ResNet

- The best accuracy now is 93.42%
- Seems like transfer learning in our case is not that promising.

Models	Accuracy
Last layer fine tune	0.8617
Full fine tune	0.9342

Other Existing Models...

Classifier	Preprocessing	Fashion test accuracy
Human Performance	Crowd-sourced evaluation of human (with no fashion expertise) performance. 1000 randomly sampled test images, 3 labels per image, majority labelling.	0.835
HOG+SVM	HOG	0.926
XgBoost	scaling the pixel values to mean=0.0 and var=1.0	0.898
2 Conv Layers with max pooling (Keras)	None	0.876
2 Conv Layers with max pooling (Tensorflow) >300 epochs	None	0.916
2 Conv Layers with max pooling and ELU activation (PyTorch)	None	0.903
2 Conv Layers net	Normalization, random horizontal flip, random vertical flip, random translation, random rotation.	0.919
2 Conv Layers with 3 FC 500K parameters	Augmentation, batch normalization	0.934
3 Conv+pooling and 2 FC+dropout	None	0.926
Capsule Network 8M parameters	Normalization and shift at most 2 pixel and horizontal flip	0.936
CNN with optional shortcuts, dense-like connectivity	standardization+augmentation+random erasing	0.947
ResNet18	Normalization, random horizontal flip, random vertical flip, random translation, random rotation.	0.949
GoogleNet with cross-entropy loss	None	0.937
Dual path network with wide resnet 28-10	standard preprocessing (mean/std subtraction/division) and augmentation (random crops/horizontal flips)	0.957
VGG16 26M parameters	None	0.935
WRN40-4 8.9M params	standard preprocessing (mean/std subtraction/division) and augmentation (random crops/horizontal flips)	0.967

Q/A

Hong Kong University of Science and Technology
Electronic & Computer Engineering
Human Language Technology Center (HLTC)

Jason WU, Peng XU, Nayeon LEE