MATH6380P Project 1
Features Extraction and Transfer Learning

PANG Hong Wing, WONG, Yik Ben

October 2020

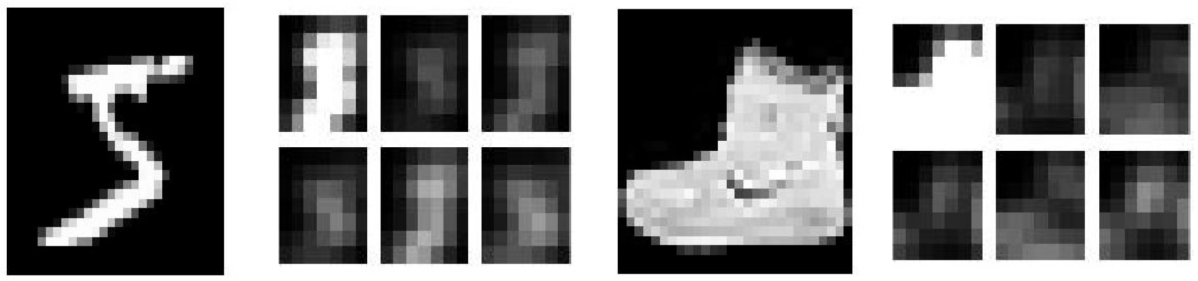
1 Introduction

In this project, features of dataset MNIST[6] and Fashion-MNIST[5] were extracted by Scattering net, VGG11 and ResNet18, and visualized by t-Distributed Stochastic Neighbor Embedding(t-SNE). Then, the performance of feature extraction method was evaluated by the index proposed in [1]. Finally, several classification algorithms were implemented to classify the class using the feature extracted and evaluate the accuracy.

2 Scattering Net

Wavelet scatting convolution neural network [2] is capable of extracting translation and rotation invariant image feature due to the deformation stability of band limited wavelet transform. The other advantage of the Scattering Net is that no training is required for this network, so both pre-train and post-train state of neural networks will be tested.

Size of [19,7,7] feature will be extracted from 28x28 image in MNIST and [64,7,7] feature for Fashion-MNIST through Scattering Net. Some of the features are shown below.



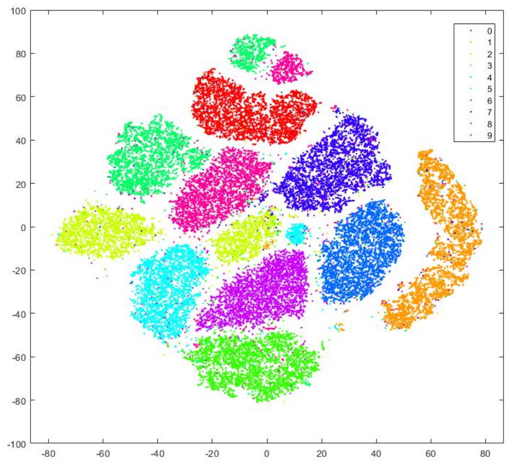
(a) Feature in MNIST

(b) Feature in Fashion-MNIST

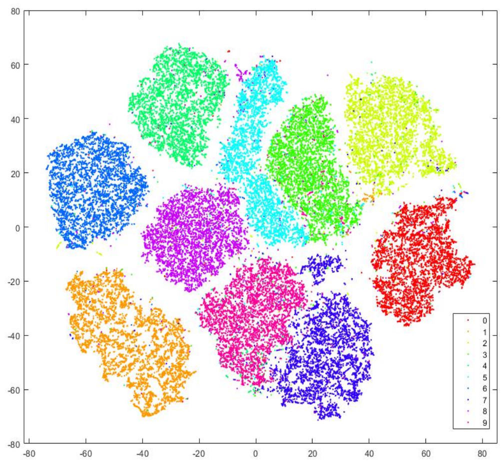
Figure 1: Feature extracted by Scattering Net.

3 Feature Visualization

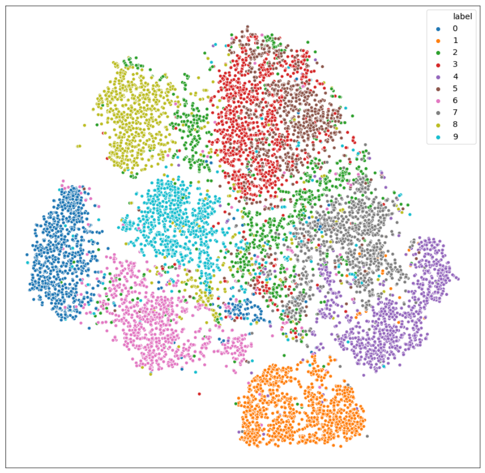
t-SNE is used to visualize the high dimensional data. Raw image, features extracted by Scattering Net, pre-train VGG11 and pre-train ResNet18 are shown below.



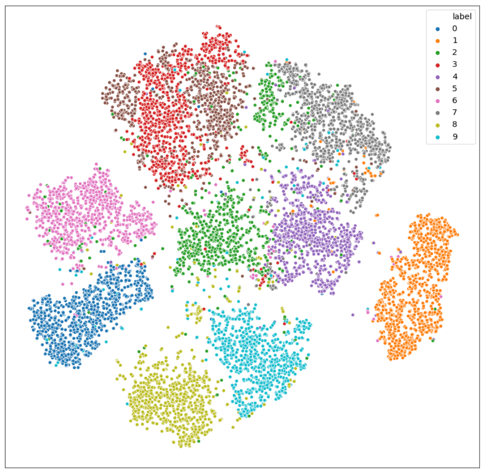
(a) Raw image



(b) Feature extracted by Scattering Net

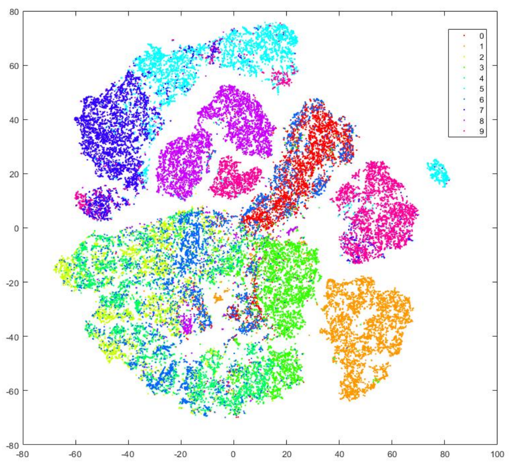


(c) Feature extracted by VGG11

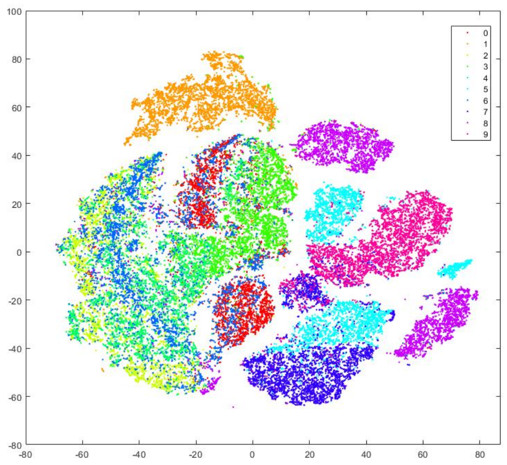


(d) Feature extracted by ResNet18

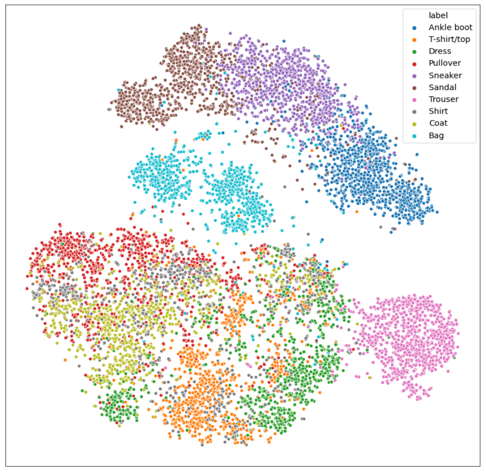
Figure 2: t-SNE visualization of 50000 images in MNIST after different feature extraction methods



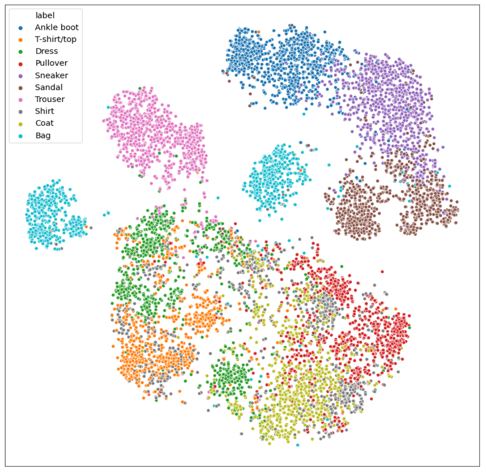
(a) Raw image



(b) Feature extracted by Scattering Net



(c) Feature extracted by VGG11



(d) Feature extracted by ResNet18

Figure 3: t-SNE visualization of 50000 images in Fashion-MNIST after different feature extraction methods

4 Feature classification

To evaluate the quality of extracted features, we compared the performance of the following classification algorithms on the raw images and various extracted features.

- Logistic regression (LR)
- Support vector machine (SVM)
- Linear discriminant analysis (LDA)
- Random forests (RF)

For VGG11 and ResNet18, one set of features were extracted using pre-trained weights on ImageNet[7], and another set were extracted following training for ten epochs on the respective dataset.

Table 1: Classification accuracy on MNIST in %

	Raw Image	Scattering Net	VGG11(pre-train)	VGG11	ResNet18(pre-train)	ResNet18
LR	89.65	97.18	95.10	99.26	95.80	99.40
SVM	**92.58**	**98.03**	**96.52**	99.22	95.65	99.45
LDA	86.05	94.44	94.48	99.12	**96.13**	**99.49**
RF	69.54	84.70	88.32	**99.28**	91.90	99.43
size	784	931	4096	4096	512	512

Table 2: Classification accuracy on Fashion-MNIST in %

	Raw Image	Scattering Net	VGG11(pre-train)	VGG11	ResNet18(pre-train)	ResNet18
LR	81.12	84.09	85.72	**94.41**	84.94	94.41
SVM	**82.62**	**85.12**	**86.42**	94.18	83.59	**94.51**
LDA	77.16	78.95	78.84	93.52	**86.00**	94.08
RF	64.28	66.65	82.04	94.34	83.54	94.34
size	784	3136	4096	4096	512	512

The performance of the Scattering Net in Fashion-MNIST is significantly worse than MNIST. A possible explanation is that Scattering Net is designed to be robust towards variance in rotations, and is better suited to handle slanted digits in MNIST. Objects in Fashion-MNIST are in the same orientation and therefore do not benefit from this advantage. Furthermore, the complexity of the class in Fashion MNIST is higher than MNIST, and deep neural networks VGG11 and ResNet18 outperform Scattering Net due to their greater representative power.

5 Neural Collapse

Donoho et. al. [1] observed that the last-layer neural activations of deep neural network architectures in general converges to the same structure as training on the network progresses, in which they named this phenomenon *neural collapse*. Quantitatively, neural collapse is indicated by the convergence of the following four measures towards zero:

Contraction of within-class covariance. The within-class covariance Σ_W, i.e. variance of feature vectors $h_{i,c}$ centered their corresponding class mean μ_c, should be relatively small compared to the between-class covariance Σ_B, which indicates the variance of class mean vectors to the global mean

μ_G. [1] suggested the following measure to represent the level of covariance dispersion as a scalar value, where $[\cdot]^+$ is the Moore-Penrose inverse:

$$\frac{\mathrm{tr}\{\Sigma_W \Sigma_B^+\}}{C} \tag{1}$$

Equinorm measure. The norm of class means should be equal. The standard deviation of class means should then converge to zero:

$$\frac{\mathrm{std}_c \|\mu_c - \mu_G\|_2}{\mathrm{avg}_c \|\mu_c - \mu_G\|_2} \tag{2}$$

Equiangularity measure. The standard deviation of cosines between all distinct pairs of class means approach zero.

$$\mathrm{std}_{c,c' \neq c}(\cos_\mu(c, c')) = \frac{\langle \mu_c - \mu_G, \mu_{c'} - \mu_G \rangle}{\|\mu_c - \mu_G\|_2 \|\mu_{c'} - \mu_G\|_2} \tag{3}$$

Maximal angle equiangularity measure. The cosines of all distinct pairs of of class means converges to $-(C-1)^{-1}$. Thus, the following measure should converge to zero:

$$\mathrm{avg}_{c,c' \neq c} |\cos_\mu(c, c') + (C-1)^{-1}| \tag{4}$$

To investigate the effect of neural collapse on transfer learning, we chose two classification architectures - VGG[3] and ResNet[4], fine-tuned the weights and compute the statistics for the training process. Specifically, we chose the smaller variants of VGG11 and ResNet18, in accordance with the choice of networks in [1].

For each model, the final fully-connected layer is replaced by a new layer with 10 neurons as output. The entire model, including pre-trained weights, are then trained for 10 epochs on each dataset. The training and test accuracy of fine-tuning the models are shown in Figure 4. Training of MNIST has reached 100% training accuracy immediately past the first epoch; Fashion-MNIST also reached near-100% accuracy by epoch 10.

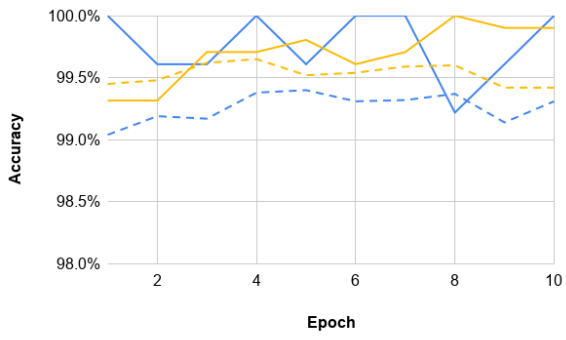
At the end of each epoch, we collected the last-layer activations of training data and compute the aforementioned measures. The results are shown in Figure 5.

We observed that most of measures indeed decrease towards zero, indicating that neural collapse also applies when training neural networks in a transfer learning setting. The only measure does not converge is the equinorm measure while training on VGG11. A possible explanation is that we have not trained long enough for this experiment; in comparison, [1] only observed convergence when training at a larger number of epochs compared to our experiment.

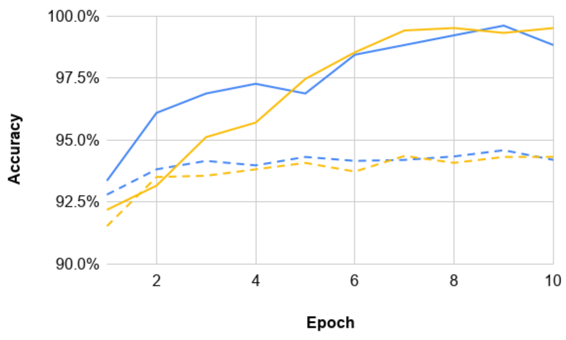
6 Contribution

PANG, Hong Wing - conduct experiment on VGG11 and ResNet18

WONG, Yik Ben - conduct experiment on Scattering Net

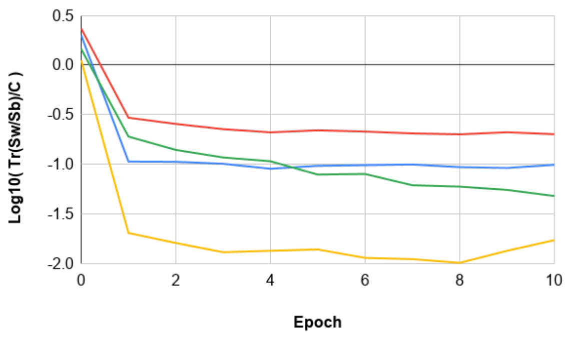


(a) Accuracy vs epoch on MNIST

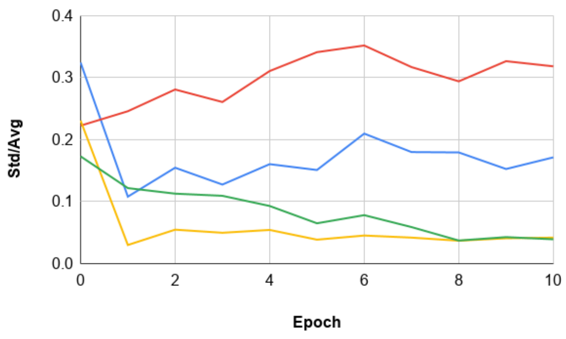


(b) Accuracy vs epoch on Fashion-MNIST

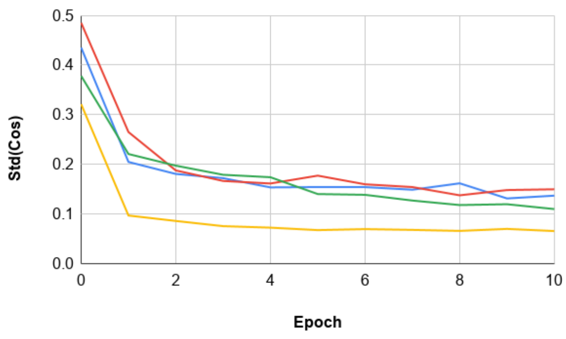
Figure 4: Accuracy(in %) for ResNet18 and VGG11 on MNIST and Fashion-MNIST. (VGG11/train-Blue, VGG11/test-dotted Blue, ResNet18/train-Yellow, ResNet18/test-dotted Yellow)



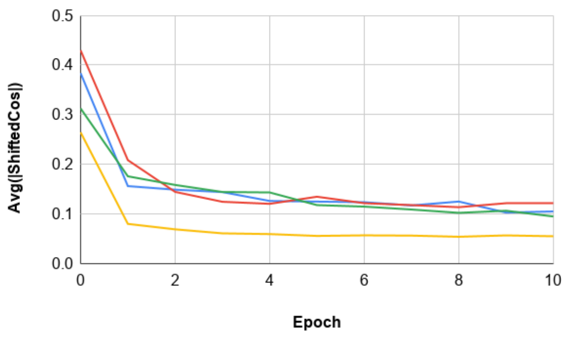
(a) Contraction of within-class covariance vs epoch



(b) Equinorm measure vs epoch



(c) Equiangularity measure vs epoch



(d) Maximal angle equiangularity measure vs epoch

Figure 5: Neural collapse result for ResNet18 and VGG11 on MNIST and Fashion-MNIST. (VGG11/MNIST-Blue, VGG11/Fashion-MNIST-Red, ResNet18/MNIST-Yellow, ResNet18/Fashion-MNIST-Green)

References

[1] V. Papyan, X. Y. Han,; D.L Donoho. Prevalence of neural collapse during the terminal phase of deep learning training. Proceedings of the National Academy of Sciences, 202015509. doi:10.1073/pnas.2015509117

[2] Mallat, Stéphane. "Group invariant Scattering." Communications on Pure and Applied Mathematics 65.10 (2012): 1331-1398.

[3] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition In *International Conference on Learning Representations (ICLR)*, 2015.

[4] K. He, X. Zhang, S. Ren and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[5] H. Xiao, K. Rasul, R. Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. https://arxiv.org/abs/1708.07747

[6] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010.

[7] Deng, J. and Dong, W. and Socher, R. and Li, L.-J. and Li, K. and Fei-Fei, L.ImageNet: A Large-Scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015.

[9] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Artificial Intelligence and Statistics (AISTATS)*, 2010.

A Implementation details

Source code to our work is available on `https://github.com/ferrophile/fentl`.

Pretrained weights - we used the models `vgg11_bn` and `resnet18` provided in the `torchvision` package in PyTorch. The former has dropout layers replaced batch normalization [8] layers, matching the experiments specified in [1].

Dataset preprocessing - as VGG11 and ResNet18 are initially designed for the ImageNet dataset, preprocessing on MNIST and Fashion-MNIST samples are required. For each sample, we resize to 224×224 to match the input size of ImageNet, then convert the grayscale image into RGB by stacking the same channel three times. We then normalize the image by the mean and s.d. of ImageNet.

Changes to architecture - for feature extraction, we removed the last fully-connected layer in ResNet18 to reveal the 512-dim. feature vector; and the last three layers in VGG11 (dropout, ReLU, fully-connected) to reveal the 4096-dim. feature vector. For fine-tuning, we replaced the last fully-connected layer in each network with a new one that has 10 output neurons. The weights of this layer is initialized as specified in [9].

Statistics calculation - the Moore-Penrose inverse is calculated using the SciPy function `pinv2` which solves the inverse from an SVD approach. To ensure numerical stability, the parameter `cond` is set to 10^{-5}, which leads to singular values less than this value discarded during construction of the inverse.