Supplementary Details for Zero-Shot Knowledge Distillation (ZSKD)

Gaurav Kumar Nayak [* 1] **Konda Reddy Mopuri** [* 2] **Vaisakh Shaj** [* 3] **R. Venkatesh Babu** [1]
Anirban Chakraborty [1]

Architecture Details Used in ZSKD

- Lenet-5 as teacher and Lenet-5-Half as student used for MNIST and Fashion-MNIST datasets

Lenet-5 Architecture (Teacher Model)	**Lenet-5-Half Architecture** (Student Model)
Layer 1: Convolution Input: 32x32x1; Output: 28x28x6 Kernel size: 5x5x1 (initialized through truncated normal with standard deviation of 0.1) No. Of Filters: 6, stride = 1 Padding = VALID Bias initialized with zeros. Activation: Relu	**Layer 1: Convolution** Input: 32x32x1; Output: 28x28x3 Kernel size: 5x5x1 (initialized through truncated normal with standard deviation of 0.1) No. Of Filters: 3, stride = 1 Padding = VALID Bias initialized with zeros. Activation: Relu
Layer 2: Pooling Max Pooling, Padding=VALID Input: 28x28x6; Output: 14x14x6	**Layer 2: Pooling** Max Pooling, Padding=VALID Input: 28x28x3; Output: 14x14x3
Layer 3: Convolution Input: 14x14x6; Output: 10x10x16 Kernel size: 5x5x6 (initialized through truncated normal with standard deviation of 0.1) No. Of Filters: 16, stride = 1 Padding = VALID Bias initialized with zeros. Activation: Relu	**Layer 3: Convolution** Input: 14x14x3; Output: 10x10x8 Kernel size: 5x5x3 (initialized through truncated normal with standard deviation of 0.1) No. Of Filters: 8, stride = 1 Padding = VALID Bias initialized with zeros. Activation: Relu
Layer 4: Pooling Max Pooling, Padding=VALID Input: 10x10x16; Output: 5x5x16	**Layer 4: Pooling** Max Pooling, Padding=VALID Input: 10x10x8; Output: 5x5x8
Flatten: Input: 5x5x16; Output=400	**Flatten:** Input: 5x5x8; Output=200
Layer 5: Fully Connected Input: 400; Output:120 Weight shape: (400,120) (initialized through truncated normal with standard deviation of 0.1) Bias initialized with zeros. Activation: Relu	**Layer 5: Fully Connected** Input: 200; Output:120 Weight shape: (200,120) (initialized through truncated normal with standard deviation of 0.1) Bias initialized with zeros. Activation: Relu

Layer 6: Fully Connected Input: 120; Output:84 Weight shape: (120,84) (initialized through truncated normal with standard deviation of 0.1) Bias initialized with zeros. Activation: Relu	**Layer 6: Fully Connected** Input: 120; Output:84 Weight shape: (120,84) (initialized through truncated normal with standard deviation of 0.1) Bias initialized with zeros. Activation: Relu
Layer 7: Fully Connected Input: 84; Output:10 Weight shape: (84,10) (initialized through truncated normal with standard deviation of 0.1) Bias initialized with zeros. Output: Logits	**Layer 7: Fully Connected** Input: 84; Output:10 Weight shape: (84,10) (initialized through truncated normal with standard deviation of 0.1) Bias initialized with zeros. Output: Logits
Layer 8: Softmax Layer	**Layer 8: Softmax Layer**

Table 1: Teacher and Student Models for MNIST and Fashion-MNIST.

- Alexnet as Teacher and Alexnet-Half as student model used for CIFAR 10 dataset

Alexnet Architecture (Teacher Model)	**Alexnet-Half Architecture** (Student Model)
Layer 1: Convolution Input: 32x32x3; Output: 32x32x48 Kernel size: 5x5x3 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 48, stride = 1 Padding = SAME Bias initialized with zeros. Activation: Relu	**Layer 1: Convolution** Input: 32x32x3; Output: 32x32x24 Kernel size: 5x5x3 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 24, stride = 1 Padding = SAME Bias initialized with zeros. Activation: Relu
Layer 2: Local Response Normalization depth_radius =2, alpha =0.0001, beta=0.75, bias=1.0	**Layer 2: Local Response Normalization** depth_radius =2, alpha =0.0001, beta=0.75, bias=1.0
Layer 3: Pooling Max Pooling, Padding=VALID Kernel size=3, stride =2 Output: 15x15x48	**Layer 3: Pooling** Max Pooling, Padding=VALID Kernel size=3, stride =2 Output: 15x15x24
Layer 4: Batch Norm	**Layer 4: Batch Norm**
Layer 5: Convolution Input: 15x15x48; Output: 15x15x128 Kernel size: 5x5x48 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 128, stride = 1 Padding = SAME Bias initialized with 1.0 Activation: Relu	**Layer 5: Convolution** Input: 15x15x24; Output: 15x15x64 Kernel size: 5x5x24 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 64, stride = 1 Padding = SAME Bias initialized with 1.0 Activation: Relu
Layer 6: Local Response Normalization depth_radius =2, alpha =0.0001, beta=0.75, bias=1.0	**Layer 6: Local Response Normalization** depth_radius =2, alpha =0.0001, beta=0.75, bias=1.0

Layer 7: Pooling Max Pooling, Padding=VALID Kernel size=3, stride =2 Output: 7x7x128	**Layer 7: Pooling** Max Pooling, Padding=VALID Kernel size=3, stride =2 Output: 7x7x64
Layer 8: Batch Norm	**Layer 8: Batch Norm**
Layer 9: Convolution Input: 7x7x128; Output: 7x7x192 Kernel size: 3x3x128 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 192, stride = 1 Padding = SAME Bias initialized with zeros. Activation: Relu	**Layer 9: Convolution** Input: 7x7x64; Output: 7x7x96 Kernel size: 3x3x64 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 96, stride = 1 Padding = SAME Bias initialized with zeros. Activation: Relu
Layer 10: Batch Norm	**Layer 10: Batch Norm**
Layer 11: Convolution Input: 7x7x192; Output: 7x7x192 Kernel size: 3x3x192 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 192, stride = 1 Padding = SAME Bias initialized with 1.0. Activation: Relu	**Layer 11: Convolution** Input: 7x7x96; Output: 7x7x96 Kernel size: 3x3x96 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 96, stride = 1 Padding = SAME Bias initialized with 1.0. Activation: Relu
Layer 12: Batch Norm	**Layer 12: Batch Norm**
Layer 13: Convolution Input: 7x7x192; Output: 7x7x128 Kernel size: 3x3x192 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 128, stride = 1 Padding = SAME Bias initialized with 1.0. Activation: Relu	**Layer 13: Convolution** Input: 7x7x96; Output: 7x7x64 Kernel size: 3x3x96 (initialized through random normal with standard deviation of 0.01) No. Of Filters: 64, stride = 1 Padding = SAME Bias initialized with 1.0. Activation: Relu
Layer 14: Pooling Max Pooling, Padding=VALID Kernel size=3, stride =2 Output: 3x3x128	**Layer 14: Pooling** Max Pooling, Padding=VALID Kernel size=3, stride =2 Output: 3x3x64
Layer 15: Batch Norm	**Layer 15: Batch Norm**
Flatten: Input: 3x3x128; Output=1152	**Flatten:** Input: 3x3x64; Output=576
Layer 16: Fully Connected Input: 1152; Output:512 Weight shape: (1152,512) (initialized through random normal with standard deviation of 0.01) Bias initialized with zeros. Activation: Relu	**Layer 16: Fully Connected** Input: 576; Output:256 Weight shape: (576,256) (initialized through random normal with standard deviation of 0.01) Bias initialized with zeros. Activation: Relu
Layer 17: Dropout Rate=0.5	**Layer 17: Dropout** Rate=0.5
Layer 18: Batch Norm	**Layer 18: Batch Norm**

Layer 19: Fully Connected Input: 512; Output:256 Weight shape: (512,256) (initialized through random normal with standard deviation of 0.01) Bias initialized with zeros. Activation: Relu	**Layer 19: Fully Connected** Input: 256; Output:128 Weight shape: (256,128) (initialized through random normal with standard deviation of 0.01) Bias initialized with zeros. Activation: Relu
Layer 20: Dropout Rate=0.5	**Layer 20: Dropout** Rate=0.5
Layer 21: Batch Norm	**Layer 21: Batch Norm**
Layer 22: Fully Connected Input: 256; Output:10 Weight shape: (256,10) (initialized through random normal with standard deviation of 0.01) Bias initialized with zeros. Output: Logits	**Layer 22: Fully Connected** Input: 128; Output:10 Weight shape: (128,10) (initialized through random normal with standard deviation of 0.01) Bias initialized with zeros. Output: Logits
Layer 23: Softmax Layer	**Layer 23: Softmax Layer**

Table 2: Teacher and Student Models for CIFAR 10.

Note: During Distillation, at train time Logits are divided by temperature of 20 and at test time the Logits are divided by temperature of 1.

Details of Hyperparameters Used in ZSKD

NOTE:- All the experiments are performed using TensorFlow framework.

1. MNIST Training

Teacher Model: Lenet-5
Student Model: Lenet-5-Half

- **Teacher Training with original data:** We take epochs as 200, batch size of 512, learning rate equal to 0.001 and Adam optimizer.

- **Student Training with original data using cross entropy loss:** Same hyperparameters as above.

- **Student Training with original data using knowledge distillation:** We take $\lambda = 0.3$ which is the weight given to cross entropy loss and the distillation loss is given the weight as 1.0. The learning rate is taken as 0.01, temperature as 20 and rest of the hyperparameters are same.

- **Data Impressions (DI) Generation:**

(a) **1% (600 DI):** Batch size of 10, number of iterations to be 1500 and learning rate as 0.1

(b) **5% (3000 DI):** Batch size as 10, number of iterations to be 1500 and learning rate as 0.1

(c) **10% (6000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 1.0

(d) **20% (12000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 2.0

(e) **40% (24000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 3.0

Student Training using DI (end to end): We take learning rate of 0.01, batch size as 512, max epochs to be 2000 and Adam optimizer.

We further finetune the model pretrained on 40% DI using mixture of DI and augmented DI samples with learning rate of 0.001

- **Class Impressions (CI) Generation:**

NOTE: We randomly sample a value (say x) from confidence range of 0.55 and 0.70. The training is done on the random noisy image till the confidence of noisy image > = confidence of x

(a) **1% (600 DI):** We take learning rate as 2.0 and student trained with learning rate of 0.01

(b) **5% (3000 DI):** We take learning rate as 0.01 and student trained with learning rate of 0.01

(c) **10% (6000 DI):** We take learning rate as 0.1 and student trained with learning rate of 0.01

(d) **20% (12000 DI):** We take learning rate as 0.01 and student trained with learning rate of 0.01

(e) **40% (24000 DI):** We take learning rate as 0.1 and student trained with learning rate of 0.001

2. Fashion - MNIST Training

Teacher Model: Lenet-5
Student Model: Lenet-5-Half

- **Teacher Training with original data:** We take epochs as 200, batch size of 512, learning rate equal to 0.001 and Adam optimizer.
- **Student Training with original data using cross entropy loss:** Same hyperparameters as above.
- **Student Training with original data using knowledge distillation:** We take $\lambda = 0.3$ which is the weight given to cross entropy loss and the distillation loss is given the weight as 1.0. The learning rate is taken as 0.01, temperature as 20 and rest of the hyperparameters are same.
- **Data Impressions (DI) Generation:**

(a) **1% (600 DI):** Batch size of 10, number of iterations to be 1500 and learning rate as 3.0

(b) **5% (3000 DI):** Batch size as 10, number of iterations to be 1500 and learning rate as 3.0

(c) **10% (6000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 1.0

(d) **20% (12000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 1.0

(e) **40% (24000 DI):** Batch size as 10, number of iterations to be 1500 and learning rate as 1.0

(f) **80% (48000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 3.0

Student Training using DI (end to end): We take batch size as 512, max epochs to be 2000 and Adam optimizer. Learning rate are taken as follows:

- Learning rate as 0.01 in case of (a), (d), (e) and (f).
- Learning rate as 0.001 in case of b).
- Learning rate as 0.0001 in case of c).

We further finetune the model pretrained on 80% DI using mixture of DI and augmented DI samples with learning rate of 0.001

- **Class Impressions (CI) Generation:**

NOTE: We randomly sample a value (say x) from confidence range of 0.55 and 0.70. The training is done on the random noisy image till the confidence of noisy image > = confidence of x

(a) **1% (600 DI):** We take learning rate as 0.01 and student trained with learning rate of 0.001

(b) **5% (3000 DI):** We take learning rate as 0.1 and student trained with learning rate of 0.001

(c) **10% (6000 DI):** We take learning rate as 2.0 and student trained with learning rate of 0.001

(d) **20% (12000 DI):** We take learning rate as 1.0 and student trained with learning rate of 0.001

(e) **40% (24000 DI):** We take learning rate as 0.01 and student trained with learning rate of 0.01

(f) **80% (48000 DI):** We take learning rate as 0.5 and student trained with learning rate of 0.001

3. CIFAR 10 Training

Teacher Model: Alexnet
Student Model: Alexnet-Half

- **Teacher Training with original data:** We take epochs as 1000, batch size of 512, learning rate equal to 0.001 and Adam optimizer.
- **Student Training with original data using cross entropy loss:** Same hyperparameters as above.
- **Student Training with original data using knowledge distillation:** We take $\lambda = 0.3$ which is the weight given to cross entropy loss and the distillation loss is given the weight as 1.0. The learning rate is taken as 0.001, temperature as 20 and rest of the hyperparameters are same.
- **Data Impressions (DI) Generation:**

(a) **1% (500 DI):** Batch size of 5, number of iterations to be 1500 and learning rate as 0.01

(b) **5% (2500 DI):** Batch size as 25, number of iterations to be 1500 and learning rate as 0.01

(c) **10% (5000 DI):** Batch size as 50, number of iterations to be 1500 and learning rate as 0.01

(d) **20% (10000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 0.01

(e) **40% (20000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 0.01

(f) **80% (40000 DI):** Batch size as 100, number of iterations to be 1500 and learning rate as 0.01

Student Training using DI (end to end): We take learning rate of 0.001, batch size as 512, max epochs to be 2000 and Adam optimizer.

We further finetune the model pretrained on 80% DI using mixture of DI and augmented DI samples with learning rate of 0.001 having batch size as 5000.

- **Class Impressions (CI) Generation:**

NOTE: We randomly sample a value (say x) from confidence range of 0.55 and 0.70. The training is done on the random noisy image till the confidence of noisy image > = confidence of x

(a) **1% (500 DI):** We take learning rate as 0.1 and student trained with learning rate of 0.001

(b) **5% (2500 DI):** We take learning rate as 0.1 and student trained with learning rate of 0.01

(c) **10% (5000 DI):** We take learning rate as 0.1 and student trained with learning rate of 0.01

(d) **20% (10000 DI):** We take learning rate as 2.0 and student trained with learning rate of 0.001

(e) **40% (20000 DI):** We take learning rate as 1.0 and student trained with learning rate of 0.001

(f) **80% (40000 DI):** We take learning rate as 0.1 and student trained with learning rate of 0.01

4. Details on Augmentation

The following operations are done on the DI's to create variety of augmented samples :-

(i) Scaling of 90%, 75% and 60% of original DI's

(ii) Translation is done on left, right, top and bottom directions by 20%

(iii) Rotation: Starts at -90° and ends at +90° to produce 10 rotated DI's such that the degree of next rotation is 20° more than the previous angle of rotation

(iv) Flipping: Operations done are flip left right, flip up down and transpose

(v) Scaling and Translation: The scaled Di's are translated on left, right, top and bottom directions by 20%

(vi) Translation and Rotation: The translated Di's are rotated

(vii) Scaling and Rotation: The scaled DI's are rotated

Below three operations are further exclusively done on the DI's extracted from Alexnet teacher model. These DI's have RGB components whereas the DI's obtained from Lenet teacher are gray scaled.

- Salt and Pepper Noise
- Gaussian Noise
- Adding Gaussian Noise to Salt and Pepper Noised DI

Ablations: With and without Augmentation

Teacher Model trained on Data set	ZSKD Performance on Student Network	
	Without Augmentation	With Augmentation
MNIST	96.98	**98.77**
Fashion MNIST	69.37	**79.62**
CIFAR 10	56.80	**69.56**

Table 3. Performance (in %) of the proposed ZSKD framework.

Uniform Prior v/s Class Similarity Prior

Dataset	Uniform Prior	Class Similarity Prior
MNIST	95.16	96.98
Fashion MNIST	56.24	69.37
Cifar 10	49.23	56.80

Table 4. Performance of proposed ZSKD (in %) using uniform and class similarity priors (without augmentation)