Project Presentation

Shujing Feng & Xinying Zou

LEHIGH UNIVERSITY

Overline

Data: MNIST, CIFAR10, Iyer

Methodology: CNN, XGBoost, Random Forest

Data Description

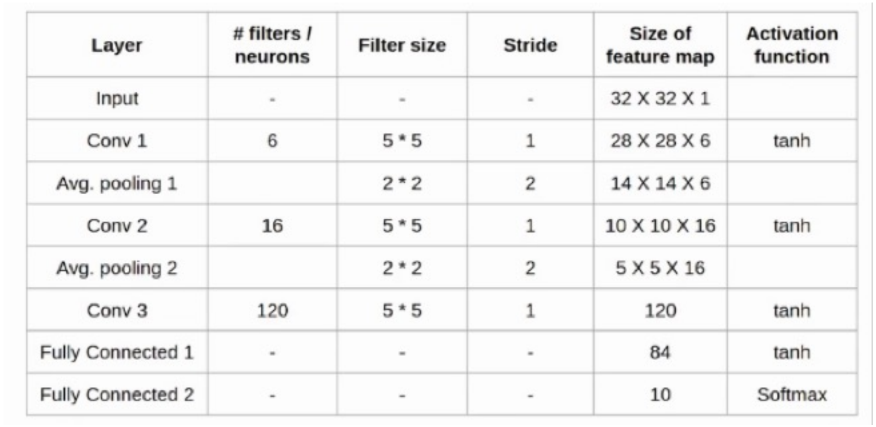
CIFAR 10: sample size:60000, 32*32 color images in 10 classes. 6000 images per class, 50000Train_data, 10000Test_data

MNIST: Handwritten dataset, 70000 example, 60000Train_data, 10000Test_data.

Iyer: Small dataset, 1*12*484

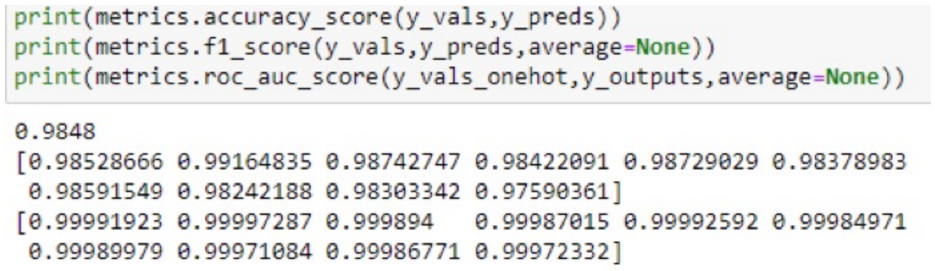
MNIST on CNN

Model:



Layer	# filters / neurons	Filter size	Stride	Size of feature map	Activation function
Input	-	-	-	32 X 32 X 1	
Conv 1	6	5 * 5	1	28 X 28 X 6	tanh
Avg. pooling 1		2 * 2	2	14 X 14 X 6	
Conv 2	16	5 * 5	1	10 X 10 X 16	tanh
Avg. pooling 2		2 * 2	2	5 X 5 X 16	
Conv 3	120	5 * 5	1	120	tanh
Fully Connected 1	-	-	-	84	tanh
Fully Connected 2	-	-	-	10	Softmax

Result:



```
print(metrics.accuracy_score(y_vals,y_preds))
print(metrics.f1_score(y_vals,y_preds,average=None))
print(metrics.roc_auc_score(y_vals_onehot,y_outputs,average=None))
```

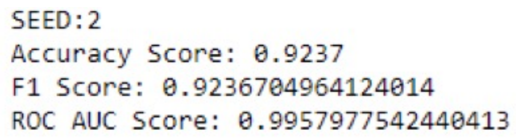
```
0.9848
[0.98528666 0.99164835 0.98742747 0.98422091 0.98729029 0.98378983
 0.98591549 0.98242188 0.98303342 0.97590361]
[0.99991923 0.99997287 0.999894   0.99987015 0.99992592 0.99984971
 0.99989979 0.99971084 0.99986771 0.99972332]
```

MNIST on XGBoost

Model:

With the defaults for XGBClassifier(
max_depth=3,
learning_rate=0.1,
n_estimators=100,
silent=True,
objective='binary:logistic',
booster='gbtree', n_jobs=1,
gamma=0,
min_child_weight=1,
max_delta_step=0,
subsample=1,
colsample_bytree=1,
colsample_bylevel=1,
reg_alpha=0,
reg_lambda=1,
scale_pos_weight=1,
base_score=0.5,
random_state=0)

Result:



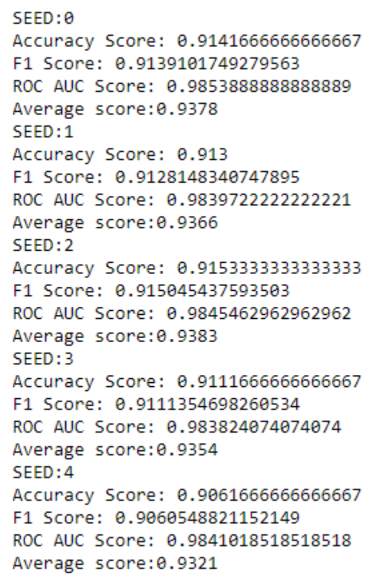
```
SEED:2
Accuracy Score: 0.9237
F1 Score: 0.9236704964124014
ROC AUC Score: 0.9957977542440413
```

MNIST on Random Forest

Model:

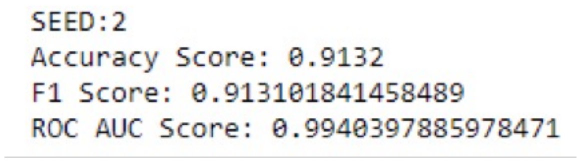
With the (random_state = 42) for RandomForestClassifier

Result of cross_validation:



```
SEED:0
Accuracy Score: 0.9141666666666667
F1 Score: 0.9139101749279563
ROC AUC Score: 0.9853888888888889
Average score:0.9378
SEED:1
Accuracy Score: 0.913
F1 Score: 0.9128148340747895
ROC AUC Score: 0.9839722222222221
Average score:0.9366
SEED:2
Accuracy Score: 0.9153333333333333
F1 Score: 0.915045437593503
ROC AUC Score: 0.9845462962962962
Average score:0.9383
SEED:3
Accuracy Score: 0.9111666666666667
F1 Score: 0.9111354698260534
ROC AUC Score: 0.983824074074074
Average score:0.9354
SEED:4
Accuracy Score: 0.9061666666666667
F1 Score: 0.9060548821152149
ROC AUC Score: 0.9841018518518518
Average score:0.9321
```

Result:



```
SEED:2
Accuracy Score: 0.9132
F1 Score: 0.913101841458489
ROC AUC Score: 0.9940397885978471
```

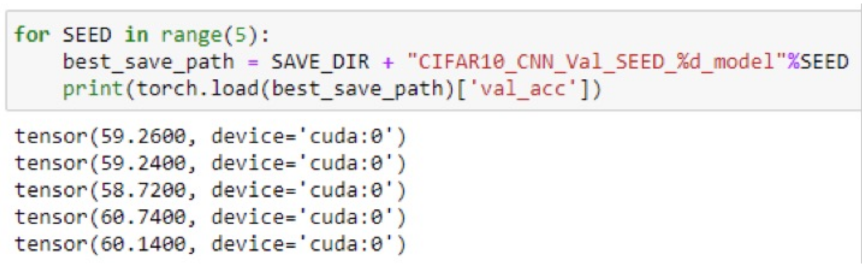
LEHIGH UNIVERSITY

CIFAR 10 on CNN

Model 1:

Layer	Filters/ neurons	Filter Size	Stride	Size of feature map	Activation function
Input				32*32*3	
Conv 1	16	5*5	1	28*28*16	tanh
Avg. pooling1		2*2	2	14*14*16	
Conv 2	32	5*5	1	10*10*32	tanh
Avg. pooling2		2*2	2	5*5*32	
Conv 3	120	5*5	1	120	tanh
Fully Connected 1				84	tanh
Fully Connected 2				10	Softmax

Result:

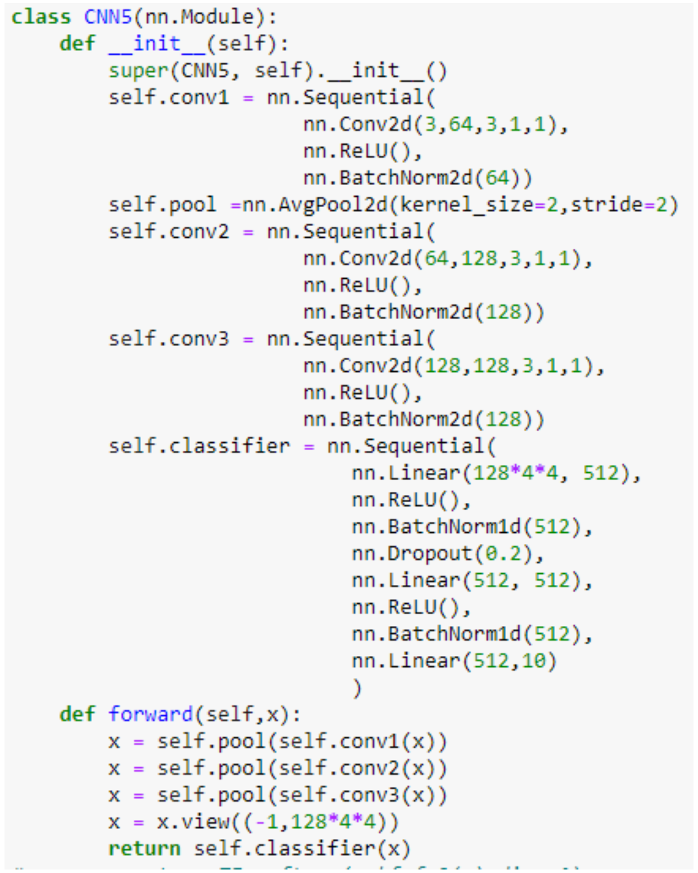


```
for SEED in range(5):
    best_save_path = SAVE_DIR + "CIFAR10_CNN_Val_SEED_%d_model"%SEED
    print(torch.load(best_save_path)['val_acc'])
```

```
tensor(59.2600, device='cuda:0')
tensor(59.2400, device='cuda:0')
tensor(58.7200, device='cuda:0')
tensor(60.7400, device='cuda:0')
tensor(60.1400, device='cuda:0')
```

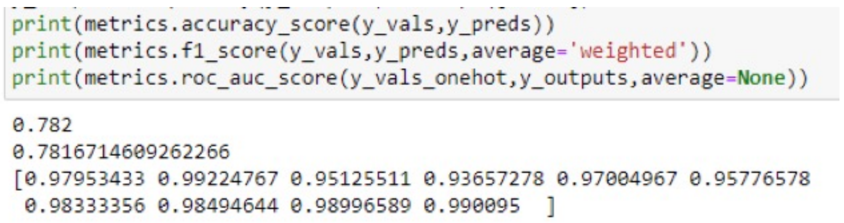
CIFAR 10 on CNN

Model 2:



```
class CNN5(nn.Module):
    def __init__(self):
        super(CNN5, self).__init__()
        self.conv1 = nn.Sequential(
                        nn.Conv2d(3,64,3,1,1),
                        nn.ReLU(),
                        nn.BatchNorm2d(64))
        self.pool =nn.AvgPool2d(kernel_size=2,stride=2)
        self.conv2 = nn.Sequential(
                        nn.Conv2d(64,128,3,1,1),
                        nn.ReLU(),
                        nn.BatchNorm2d(128))
        self.conv3 = nn.Sequential(
                        nn.Conv2d(128,128,3,1,1),
                        nn.ReLU(),
                        nn.BatchNorm2d(128))
        self.classifier = nn.Sequential(
                            nn.Linear(128*4*4, 512),
                            nn.ReLU(),
                            nn.BatchNorm1d(512),
                            nn.Dropout(0.2),
                            nn.Linear(512, 512),
                            nn.ReLU(),
                            nn.BatchNorm1d(512),
                            nn.Linear(512,10)
                            )
    def forward(self,x):
        x = self.pool(self.conv1(x))
        x = self.pool(self.conv2(x))
        x = self.pool(self.conv3(x))
        x = x.view((-1,128*4*4))
        return self.classifier(x)
```

Result:



```
print(metrics.accuracy_score(y_vals,y_preds))
print(metrics.f1_score(y_vals,y_preds,average='weighted'))
print(metrics.roc_auc_score(y_vals_onehot,y_outputs,average=None))
```

```
0.782
0.7816714609262266
[0.97953433 0.99224767 0.95125511 0.93657278 0.97004967 0.95776578
 0.98333356 0.98494644 0.98996589 0.990095   ]
```

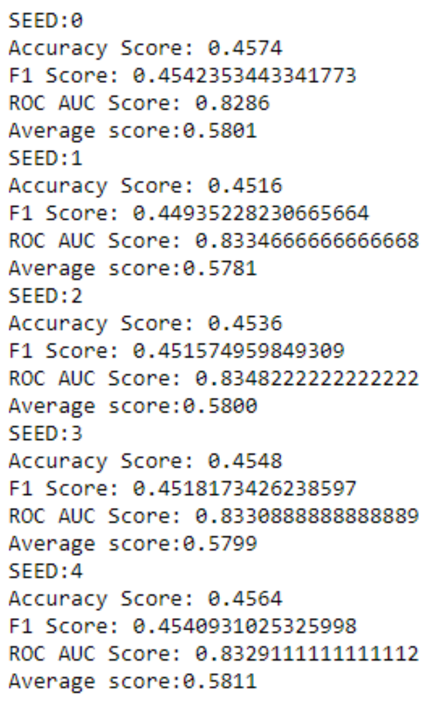
LEHIGH UNIVERSITY

CIFAR 10 on XGBoost

Model:

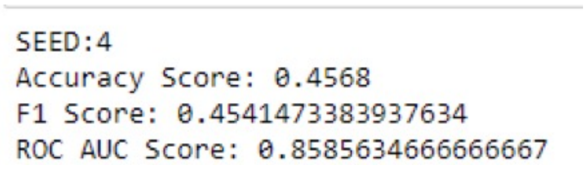
With the defaults for XGBClassifie

Result of cross validation:



```
SEED:0
Accuracy Score: 0.4574
F1 Score: 0.4542353443341773
ROC AUC Score: 0.8286
Average score:0.5801
SEED:1
Accuracy Score: 0.4516
F1 Score: 0.44935228230665664
ROC AUC Score: 0.8334666666666668
Average score:0.5781
SEED:2
Accuracy Score: 0.4536
F1 Score: 0.451574959849309
ROC AUC Score: 0.8348222222222222
Average score:0.5800
SEED:3
Accuracy Score: 0.4548
F1 Score: 0.4518173426238597
ROC AUC Score: 0.8330888888888889
Average score:0.5799
SEED:4
Accuracy Score: 0.4564
F1 Score: 0.4540931025325998
ROC AUC Score: 0.8329111111111112
Average score:0.5811
```

Result:



```
SEED:4
Accuracy Score: 0.4568
F1 Score: 0.4541473383937634
ROC AUC Score: 0.8585634666666667
```

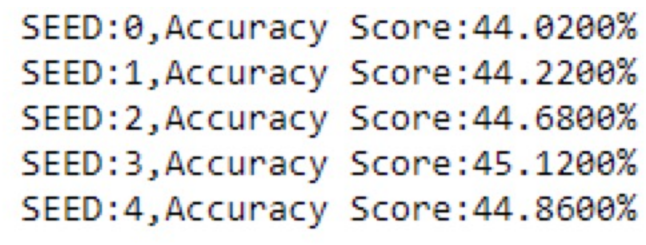
Bad performance!

CIFAR 10 on Random Forest

Model:

With the random_state = 42 for RandomForestClassifier

Result of cross_validation:



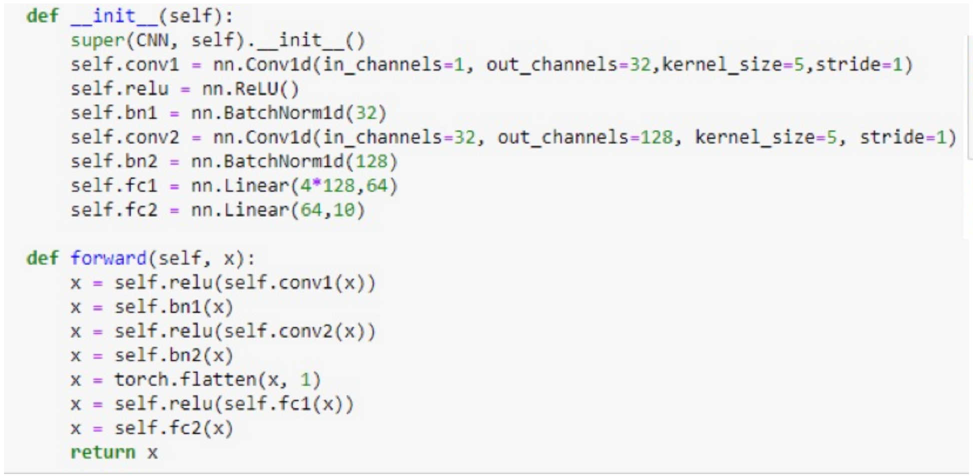
```
SEED:0,Accuracy Score:44.0200%
SEED:1,Accuracy Score:44.2200%
SEED:2,Accuracy Score:44.6800%
SEED:3,Accuracy Score:45.1200%
SEED:4,Accuracy Score:44.8600%
```

Result:

Bad performance!

Iyer on CNN

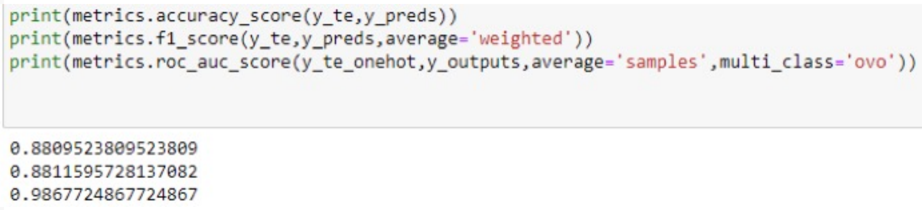
Model:



```
def __init__(self):
    super(CNN, self).__init__()
    self.conv1 = nn.Conv1d(in_channels=1, out_channels=32,kernel_size=5,stride=1)
    self.relu = nn.ReLU()
    self.bn1 = nn.BatchNorm1d(32)
    self.conv2 = nn.Conv1d(in_channels=32, out_channels=128, kernel_size=5, stride=1)
    self.bn2 = nn.BatchNorm1d(128)
    self.fc1 = nn.Linear(4*128,64)
    self.fc2 = nn.Linear(64,10)

def forward(self, x):
    x = self.relu(self.conv1(x))
    x = self.bn1(x)
    x = self.relu(self.conv2(x))
    x = self.bn2(x)
    x = torch.flatten(x, 1)
    x = self.relu(self.fc1(x))
    x = self.fc2(x)
    return x
```

Result:



```
print(metrics.accuracy_score(y_te,y_preds))
print(metrics.f1_score(y_te,y_preds,average='weighted'))
print(metrics.roc_auc_score(y_te_onehot,y_outputs,average='samples',multi_class='ovo'))
```

```
0.8809523809523809
0.8811595728137082
0.9867724867724867
```

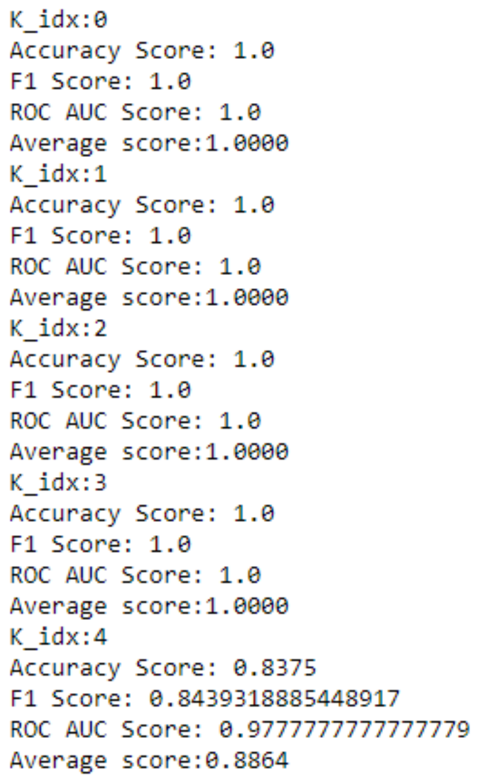
LEHIGH UNIVERSITY

Iyer on XGBoost

Model:

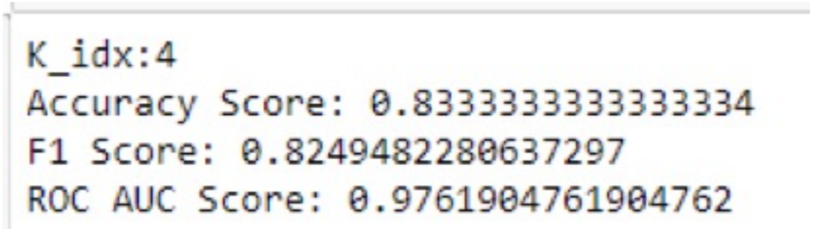
With the defaults for XGBClassifie

Result of cross validation:



```
K_idx:0
Accuracy Score: 1.0
F1 Score: 1.0
ROC AUC Score: 1.0
Average score:1.0000
K_idx:1
Accuracy Score: 1.0
F1 Score: 1.0
ROC AUC Score: 1.0
Average score:1.0000
K_idx:2
Accuracy Score: 1.0
F1 Score: 1.0
ROC AUC Score: 1.0
Average score:1.0000
K_idx:3
Accuracy Score: 1.0
F1 Score: 1.0
ROC AUC Score: 1.0
Average score:1.0000
K_idx:4
Accuracy Score: 0.8375
F1 Score: 0.8439318885448917
ROC AUC Score: 0.9777777777777779
Average score:0.8864
```

Result:



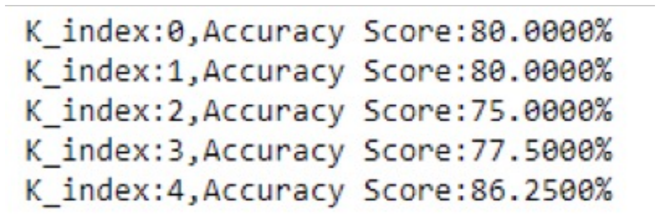
```
K_idx:4
Accuracy Score: 0.8333333333333334
F1 Score: 0.8249482280637297
ROC AUC Score: 0.9761904761904762
```

Iyer on Random Forest

Model:

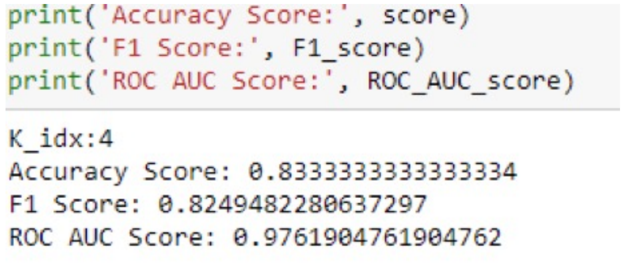
With the random_state = 42 for RandomForestClassifier

Result of cross_validation:



```
K_index:0,Accuracy Score:80.0000%
K_index:1,Accuracy Score:80.0000%
K_index:2,Accuracy Score:75.0000%
K_index:3,Accuracy Score:77.5000%
K_index:4,Accuracy Score:86.2500%
```

Result:



```
print('Accuracy Score:', score)
print('F1 Score:', F1_score)
print('ROC AUC Score:', ROC_AUC_score)
```

```
K_idx:4
Accuracy Score: 0.8333333333333334
F1 Score: 0.8249482280637297
ROC AUC Score: 0.9761904761904762
```

LEHIGH

UNIVERSITY