sign_language_cnn_100_percent

May 5, 2020

```
In [2]: import matplotlib.pyplot as plt
        import numpy as np
        import tensorflow as tf
        import pandas as pd
        from tensorflow import keras
        from tensorflow.keras.layers import Conv2D, Flatten, MaxPooling2D, AveragePooling2D, Den
        from keras.preprocessing.image import ImageDataGenerator
        import string

        from libitmal import kernelfuns as itmalkernelfuns
        itmalkernelfuns.EnableGPU()

        %matplotlib inline

        def get_mnist_dataset():

            test_pd = pd.read_csv("./SignLanguageData/sign_mnist_test.csv",
                skiprows=1)
            train_pd = pd.read_csv("./SignLanguageData/sign_mnist_train.csv",
                skiprows=1)

            return train_pd, test_pd

        train_pd, test_pd = get_mnist_dataset()
        X_train, X_test = train_pd.values[:,1:], test_pd.values[:,1:]
        y_train, y_test = train_pd.values[:,0], test_pd.values[:,0]

        class_names = list(string.ascii_lowercase)

        train_pd.head()
```

```
Out[2]:     3  107  118  127  134  139  143  146  150  153  ...  207.4  207.5  207.6  \
        0   6  155  157  156  156  156  157  156  158  158  ...     69    149    128
        1   2  187  188  188  187  187  186  187  188  187  ...    202    201    200
        2   2  211  211  212  212  211  210  211  210  210  ...    235    234    233
        3  13  164  167  170  172  176  179  180  184  185  ...     92    105    105
        4  16  161  168  172  173  178  184  189  193  196  ...     76     74     68
```

```
   207.7  206.4  206.5  206.6  204.6  203.8  202.13
0     87     94    163    175    103    135     149
1    199    198    199    198    195    194     195
2    231    230    226    225    222    229     163
3    108    133    163    157    163    164     179
4     62     53     55     48    238    255     255

[5 rows x 785 columns]
```

```
In [3]: X_train = X_train / 255
        X_test =  X_test / 255

        X_train = X_train.reshape(*X_train.shape[:1], 28, 28)
        X_test = X_test.reshape(*X_test.shape[:1], 28, 28)

        X_train = X_train.reshape(X_train.shape[0], 28, 28, 1)
        X_test = X_test.reshape(X_test.shape[0], 28, 28, 1)

        batch_size, height, width, channel = X_train.shape

        print(X_train.shape)
```

```
(27454, 28, 28, 1)
```

```
In [185]: # Datageneration form https://www.kaggle.com/madz2000/cnn-using-keras-99-7-accuracy
          datagen = ImageDataGenerator(
                  featurewise_center=False,  # set input mean to 0 over the dataset
                  samplewise_center=False,  # set each sample mean to 0
                  featurewise_std_normalization=False,  # divide inputs by std of the dataset
                  samplewise_std_normalization=False,  # divide each input by its std
                  zca_whitening=False,  # apply ZCA whitening
                  rotation_range=10,  # randomly rotate images in the range (degrees, 0 to 180)
                  zoom_range = 0.1, # Randomly zoom image
                  width_shift_range=0.1,  # randomly shift images horizontally (fraction of tota
                  height_shift_range=0.1,  # randomly shift images vertically (fraction of total
                  horizontal_flip=False,  # randomly flip images
                  vertical_flip=False # randomly flip images
          )


          datagen.fit(X_train)
```

```
In [6]: def create_le_net():
            model = keras.models.Sequential([
                ZeroPadding2D(input_shape=X_train.shape[1:], padding=(3, 3)),
                Conv2D(filters=6, kernel_size=(5, 5), strides=1, activation="tanh"),
                AveragePooling2D(pool_size=6, strides=2, padding="same"),
                Conv2D(filters=16, kernel_size=(5, 5), strides=1, activation="tanh"),
                AveragePooling2D(pool_size=6, strides=2, padding="same"),
                Conv2D(filters=120, kernel_size=(5, 5), strides=1, activation="tanh"),
                Flatten(),
                Dense(84, activation="tanh"),
                Dense(len(class_names), activation="softmax")
            ])

            return model

        def create_model():
            model = keras.models.Sequential([
                ZeroPadding2D(input_shape=X_train.shape[1:], padding=(3, 3)),
                Conv2D(filters=16, kernel_size=(3, 3), activation="relu"),
                AveragePooling2D(),
                Conv2D(filters=32, kernel_size=(3, 3), activation="relu"),
                AveragePooling2D(),
                Flatten(),
                Dense(256, activation="relu"),
                Dense(512, activation="relu"),
                Dense(128, activation="relu"),
                Dense(64, activation="relu"),
                Dense(len(class_names), activation="softmax")
            ])

            return model
```

In [201]:
```
X_test.shape
```

Out[201]:
```
(7171, 28, 28, 1)
```

In [187]:
```
# LeNet
model = create_model()
```

In [188]:
```
#keras.utils.plot_model(model, "my_mnist_model.png", show_shapes=True)
```

In [189]:
```
model.compile(loss="sparse_categorical_crossentropy",
              optimizer="adam",
              metrics=["accuracy"])
```

In [190]:
```
from keras.callbacks import EarlyStopping, ModelCheckpoint

early_stopping = EarlyStopping(monitor='loss',
                               patience=30,
                               verbose=0,
                               mode='min')

mcp_save = ModelCheckpoint('cnn_model_checkpoint.h5',
                           save_best_only=True,
                           monitor='val_loss',
                           mode='min')

history = model.fit(datagen.flow(X_train,
                                 y_train,
                                 batch_size = 128,
                                 validation_split=0.2),
                        epochs=500,
                        validation_split=0.2,
                        callbacks=[early_stopping, mcp_save])
```

```
Epoch 1/500
215/215 [==============================] - 5s 22ms/step - loss: 2.6089 - acc: 0.1943
Epoch 2/500
215/215 [==============================] - 4s 19ms/step - loss: 1.4556 - acc: 0.5055
Epoch 3/500
215/215 [==============================] - 4s 20ms/step - loss: 0.8565 - acc: 0.7064
Epoch 4/500
215/215 [==============================] - 4s 19ms/step - loss: 0.5162 - acc: 0.8204
Epoch 5/500
215/215 [==============================] - 4s 20ms/step - loss: 0.3500 - acc: 0.8827
Epoch 6/500
215/215 [==============================] - 4s 20ms/step - loss: 0.2634 - acc: 0.9108
Epoch 7/500
215/215 [==============================] - 4s 20ms/step - loss: 0.1906 - acc: 0.9372
Epoch 8/500
215/215 [==============================] - 4s 20ms/step - loss: 0.1396 - acc: 0.9549
Epoch 9/500
215/215 [==============================] - 4s 20ms/step - loss: 0.1299 - acc: 0.9565
Epoch 10/500
215/215 [==============================] - 4s 20ms/step - loss: 0.1100 - acc: 0.9629
Epoch 11/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0986 - acc: 0.9682
Epoch 12/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0891 - acc: 0.9703
Epoch 13/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0690 - acc: 0.9774
Epoch 14/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0637 - acc: 0.9788
Epoch 15/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0568 - acc: 0.9819
Epoch 16/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0543 - acc: 0.9820
Epoch 17/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0496 - acc: 0.9832
Epoch 18/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0553 - acc: 0.9820
Epoch 19/500
```

```
215/215 [==============================] - 4s 20ms/step - loss: 0.0540 - acc: 0.9825
Epoch 20/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0378 - acc: 0.9880
Epoch 21/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0424 - acc: 0.9862
Epoch 22/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0346 - acc: 0.9890
Epoch 23/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0317 - acc: 0.9896
Epoch 24/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0388 - acc: 0.9872
Epoch 25/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0342 - acc: 0.9889
Epoch 26/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0291 - acc: 0.9902
Epoch 27/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0288 - acc: 0.9909
Epoch 28/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0269 - acc: 0.9907
Epoch 29/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0291 - acc: 0.9902
Epoch 30/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0250 - acc: 0.9924
Epoch 31/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0297 - acc: 0.9907
Epoch 32/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0252 - acc: 0.9920
Epoch 33/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0292 - acc: 0.9910
Epoch 34/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0274 - acc: 0.9913
Epoch 35/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0257 - acc: 0.9916
Epoch 36/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0188 - acc: 0.9941
Epoch 37/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0178 - acc: 0.9943
Epoch 38/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0195 - acc: 0.9943
Epoch 39/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0242 - acc: 0.9924
Epoch 40/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0182 - acc: 0.9942
Epoch 41/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0204 - acc: 0.9931
Epoch 42/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0194 - acc: 0.9939
Epoch 43/500
```

```
215/215 [==============================] - 4s 19ms/step - loss: 0.0181 - acc: 0.9940
Epoch 44/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0215 - acc: 0.9927
Epoch 45/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0146 - acc: 0.9956
Epoch 46/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0201 - acc: 0.9932
Epoch 47/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0205 - acc: 0.9939
Epoch 48/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0153 - acc: 0.9947
Epoch 49/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0164 - acc: 0.9948
Epoch 50/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0140 - acc: 0.9962
Epoch 51/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0134 - acc: 0.9957
Epoch 52/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0177 - acc: 0.9944
Epoch 53/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0165 - acc: 0.9946
Epoch 54/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0212 - acc: 0.9937
Epoch 55/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0111 - acc: 0.9964
Epoch 56/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0148 - acc: 0.9956
Epoch 57/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0177 - acc: 0.9944
Epoch 58/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0153 - acc: 0.9956
Epoch 59/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0062 - acc: 0.9982
Epoch 60/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0112 - acc: 0.9964
Epoch 61/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0269 - acc: 0.9916
Epoch 62/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0085 - acc: 0.9972
Epoch 63/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0167 - acc: 0.9948
Epoch 64/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0083 - acc: 0.9972
Epoch 65/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0125 - acc: 0.9962
Epoch 66/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0100 - acc: 0.9969
Epoch 67/500
```

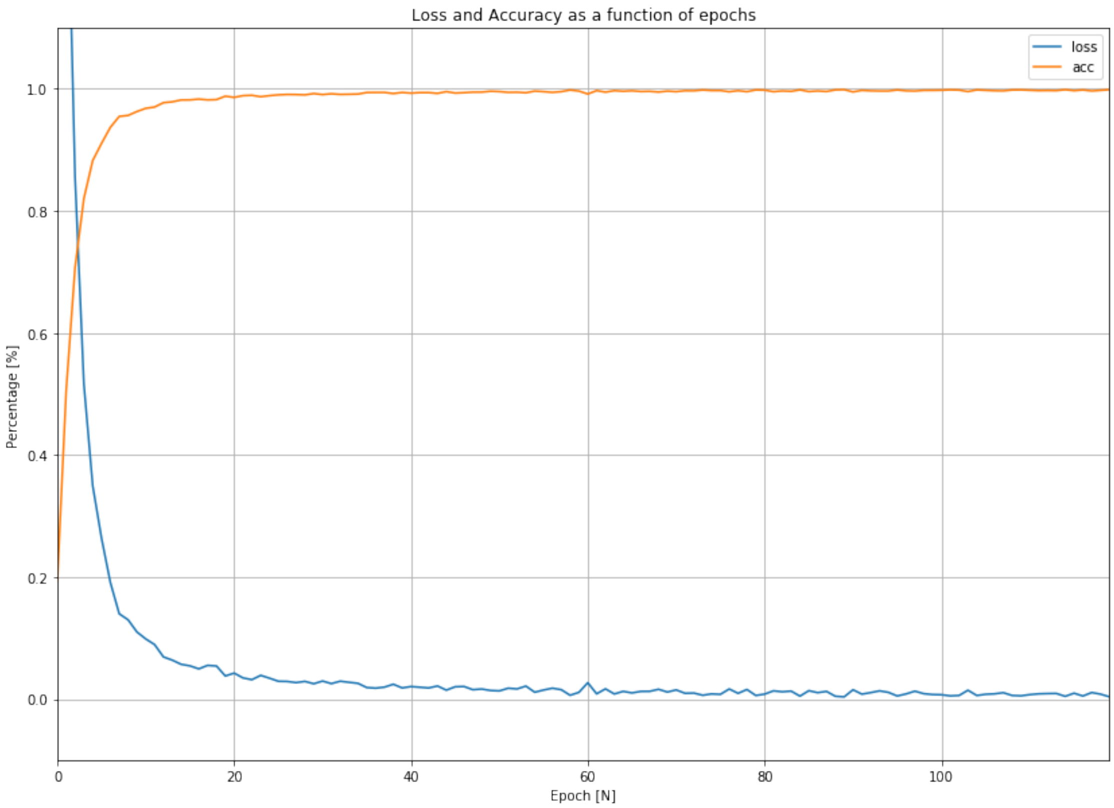
```
215/215 [==============================] - 4s 20ms/step - loss: 0.0125 - acc: 0.9958
Epoch 68/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0127 - acc: 0.9960
Epoch 69/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0160 - acc: 0.9949
Epoch 70/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0116 - acc: 0.9964
Epoch 71/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0149 - acc: 0.9956
Epoch 72/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0094 - acc: 0.9970
Epoch 73/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0098 - acc: 0.9970
Epoch 74/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0062 - acc: 0.9983
Epoch 75/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0084 - acc: 0.9974
Epoch 76/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0077 - acc: 0.9974
Epoch 77/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0165 - acc: 0.9954
Epoch 78/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0093 - acc: 0.9971
Epoch 79/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0156 - acc: 0.9955
Epoch 80/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0058 - acc: 0.9984
Epoch 81/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0080 - acc: 0.9980
Epoch 82/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0134 - acc: 0.9955
Epoch 83/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0119 - acc: 0.9967
Epoch 84/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0130 - acc: 0.9960
Epoch 85/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0047 - acc: 0.9985
Epoch 86/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0137 - acc: 0.9957
Epoch 87/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0104 - acc: 0.9966
Epoch 88/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0125 - acc: 0.9957
Epoch 89/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0045 - acc: 0.9984
Epoch 90/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0035 - acc: 0.9988
Epoch 91/500
```

```
215/215 [==============================] - 4s 19ms/step - loss: 0.0152 - acc: 0.9951
Epoch 92/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0080 - acc: 0.9977
Epoch 93/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0105 - acc: 0.9969
Epoch 94/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0134 - acc: 0.9966
Epoch 95/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0112 - acc: 0.9965
Epoch 96/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0050 - acc: 0.9984
Epoch 97/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0084 - acc: 0.9969
Epoch 98/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0129 - acc: 0.9965
Epoch 99/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0088 - acc: 0.9977
Epoch 100/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0074 - acc: 0.9977
Epoch 101/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0070 - acc: 0.9979
Epoch 102/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0052 - acc: 0.9985
Epoch 103/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0059 - acc: 0.9981
Epoch 104/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0144 - acc: 0.9957
Epoch 105/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0058 - acc: 0.9985
Epoch 106/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0078 - acc: 0.9977
Epoch 107/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0085 - acc: 0.9971
Epoch 108/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0105 - acc: 0.9968
Epoch 109/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0058 - acc: 0.9983
Epoch 110/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0053 - acc: 0.9985
Epoch 111/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0072 - acc: 0.9979
Epoch 112/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0085 - acc: 0.9973
Epoch 113/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0094 - acc: 0.9974
Epoch 114/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0091 - acc: 0.9973
Epoch 115/500
```

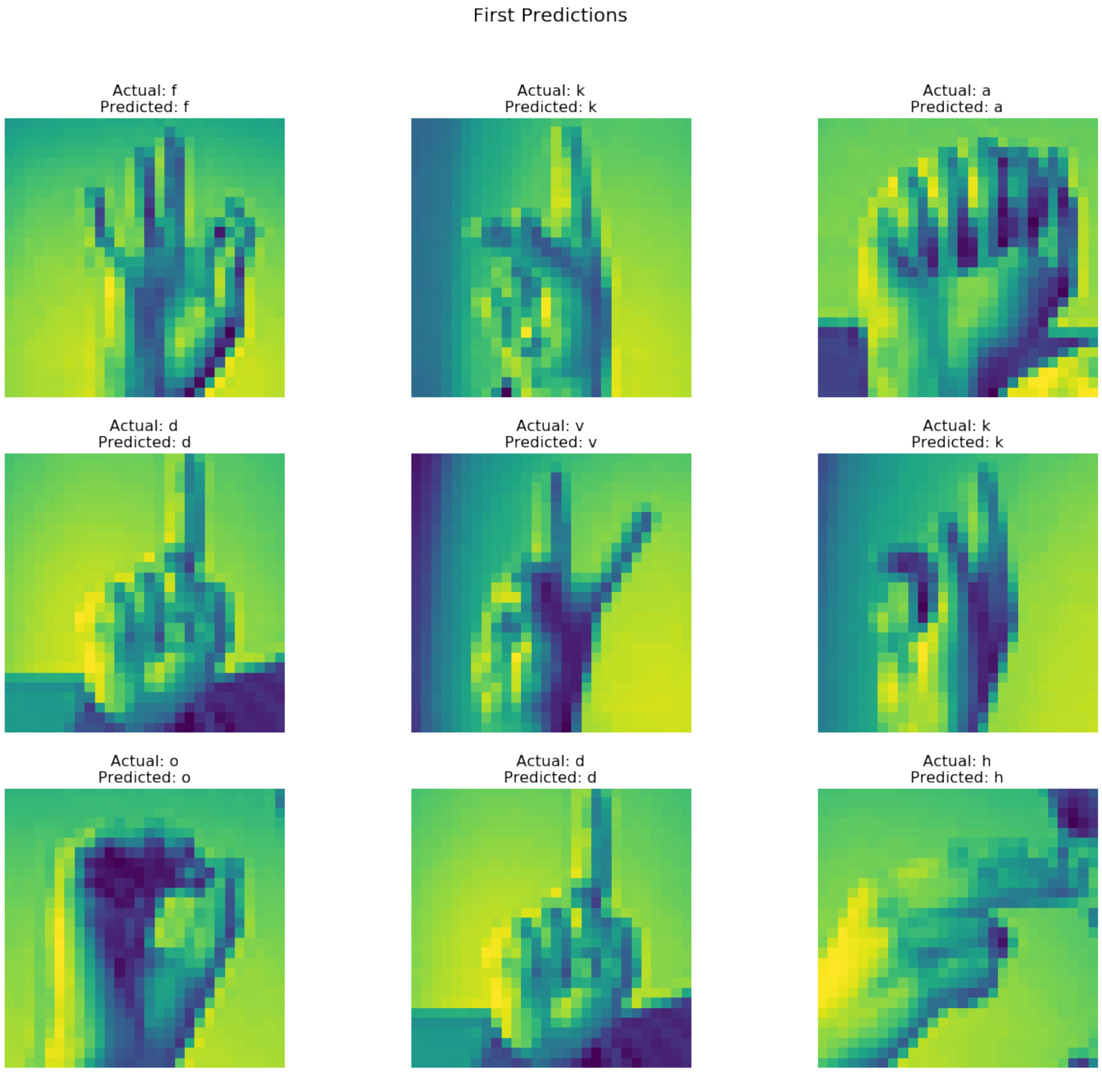
```
215/215 [==============================] - 4s 20ms/step - loss: 0.0044 - acc: 0.9988
Epoch 116/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0095 - acc: 0.9972
Epoch 117/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0049 - acc: 0.9983
Epoch 118/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0106 - acc: 0.9967
Epoch 119/500
215/215 [==============================] - 4s 20ms/step - loss: 0.0080 - acc: 0.9977
Epoch 120/500
215/215 [==============================] - 4s 19ms/step - loss: 0.0036 - acc: 0.9988
```

```
In [191]: import pandas as pd

          evaluation = model.evaluate(X_test, y_test)
          print(f'Model evaluation: {evaluation}')

          pd.DataFrame(history.history).plot(figsize=(14, 10))
          plt.grid(True)
          plt.gca().set_ylim(-0.1, 1.1)
          plt.xlabel("Epoch [N]")
          plt.ylabel("Percentage [%]")
          plt.title("Loss and Accuracy as a function of epochs")
          plt.show()
```

```
7171/7171 [==============================] - 1s 71us/step
Model evaluation: [0.0008896882954271737, 1.0]
```

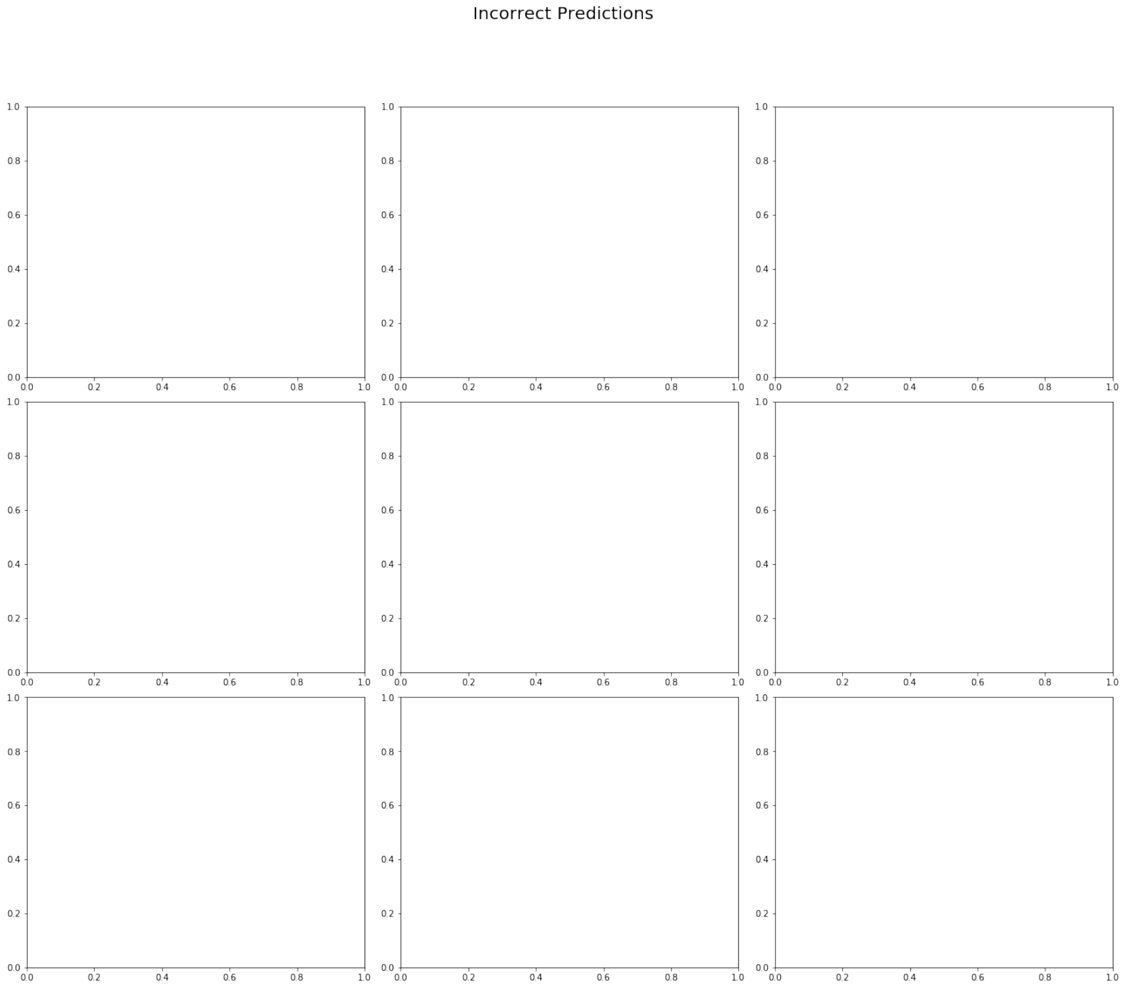


```
In [192]: import math

          num_rows = 3
          num_cols = 3

          X_new = X_test[:num_rows*num_cols]

          y_pred = model.predict_classes(X_new)

          fig, ax = plt.subplots(num_rows, num_cols, figsize=(18, 16))
          for index, image in enumerate(X_new):
              ax[math.floor(index/num_rows), index%num_rows].imshow(image.reshape((28,28)))
              ax[math.floor(index/num_rows), index%num_rows].set_title(
                  f"Actual: {class_names[y_test[index]]}\nPredicted: {class_names[y_pred[index]]
                  fontsize=16)
              ax[int(index/num_rows), index%num_rows].axis('off')

          fig.tight_layout()
          fig.suptitle(f'First Predictions', fontsize=20)
          fig.subplots_adjust(top=0.88)
          fig.show()
```



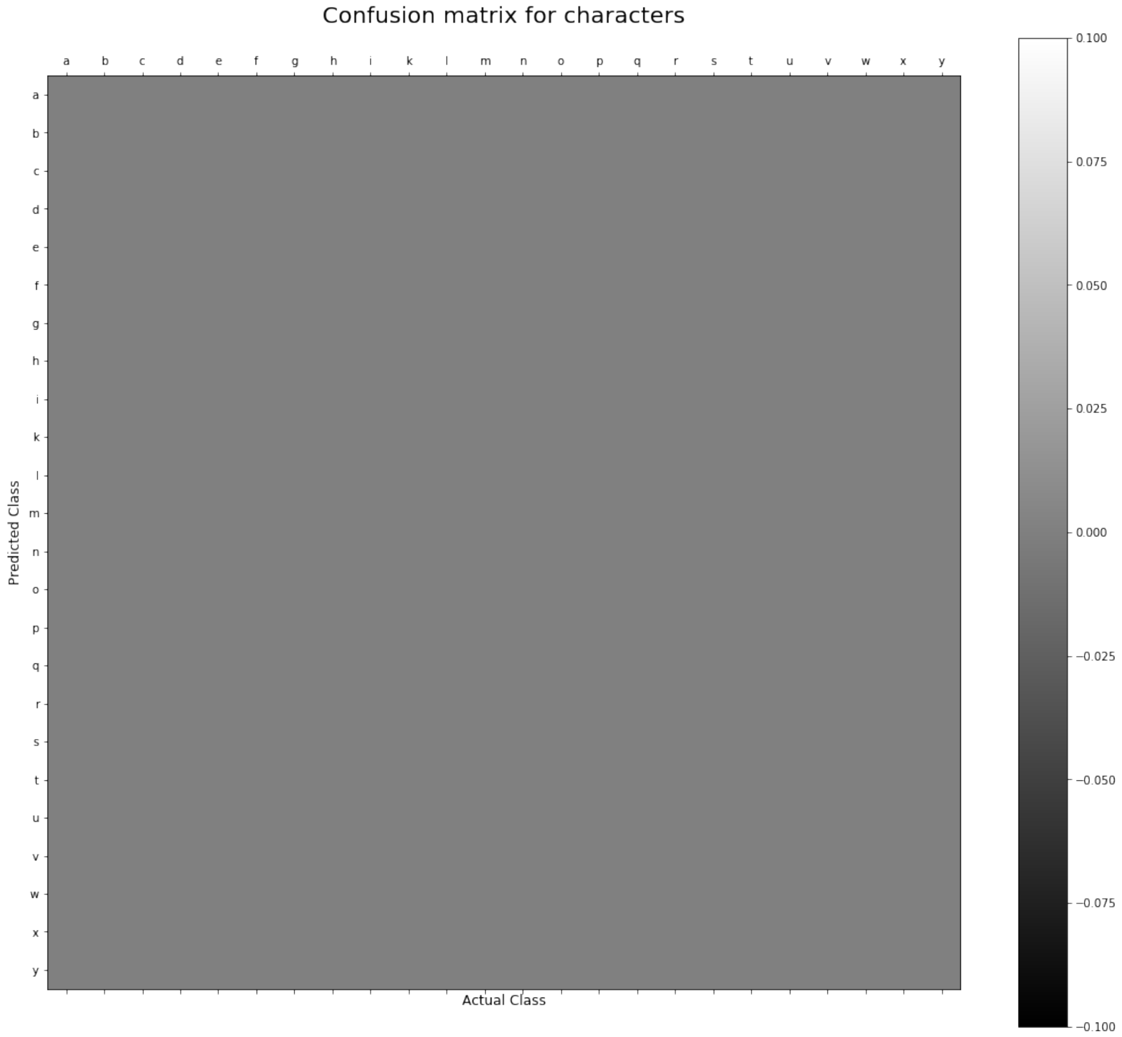
```
In [193]: y_pred = model.predict_classes(X_test)

          confusion_indices = np.where(y_pred != y_test)
          X_confusion = X_test[confusion_indices]
          y_pred_confusion = y_pred[confusion_indices]
          y_test_confusion = y_test[confusion_indices]

          fig, ax = plt.subplots(num_rows, num_cols, figsize=(18, 16))
          for index, image in enumerate(X_confusion[:num_rows*num_cols]):
              ax[math.floor(index/num_rows), index%num_rows].imshow(image.reshape((28,28)), inte
              ax[math.floor(index/num_rows), index%num_rows].set_title(
                  f"Actual: {class_names[y_test_confusion[index]]}\nPredicted: {class_names[y_pr
                  fontsize=16)
              ax[int(index/num_rows), index%num_rows].axis('off')
```

```
fig.tight_layout()
fig.suptitle('Incorrect Predictions', fontsize=20)
fig.subplots_adjust(top=0.88)
fig.show()
```



```
In [194]: import sklearn.metrics as metrics

          confusion_matrix = metrics.confusion_matrix(y_test, y_pred)
          row_sum = confusion_matrix.sum(axis=1, keepdims=True)
          norm_confusion_matrix = confusion_matrix / row_sum

          # Becausse j and z aren't possible we cant include them in confusion matrix
          class_names_clean = class_names.copy()
          class_names_clean.remove('j')
          class_names_clean.remove('z')
```

```
np.fill_diagonal(norm_confusion_matrix, 0)

fig, ax = plt.subplots(figsize=(18, 16))

mat_ax = ax.matshow(norm_confusion_matrix, interpolation='nearest', cmap=plt.cm.gray)
fig.colorbar(mat_ax)
ax.set_title('Confusion matrix for characters', fontsize=20)
ax.set_xlabel('Actual Class', fontsize=12)
ax.set_ylabel('Predicted Class', fontsize=12)
ax.set_xticks(ticks=np.arange(0, len(class_names_clean)))
ax.set_xticklabels(class_names_clean)
ax.set_yticks(ticks=np.arange(0, len(class_names_clean)))
ax.set_yticklabels(class_names_clean)
fig.show()
```



```
In [195]: y_pred = model.predict_classes(X_test)
          print(metrics.f1_score(y_test, y_pred, average="micro"))
```

```
1.0
```

```
In [196]: model.save('cnn_model_100.h5')
```