aslExtractedFeaturesTFGNotebook

April 24, 2021

[1]:
```
import os
import numpy as np
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
from keras.layers import *
from keras import optimizers, callbacks
import keras.backend as k

%matplotlib inline
import matplotlib.pyplot as plt

import sys, math
import pandas as pd
from sklearn import preprocessing
```

```
Using TensorFlow backend.
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:516: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:517: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:518: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:519: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:520: FutureWarning: Passing
```

```
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorflow/python/framework/dtypes.py:525: FutureWarning: Passing
(type, 1) or '1type' as a synonym of type is deprecated; in a future version of
numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:541: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:542: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:543: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:544: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:545: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/tensorboard/compat/tensorflow_stub/dtypes.py:550: FutureWarning:
Passing (type, 1) or '1type' as a synonym of type is deprecated; in a future
version of numpy, it will be understood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

[2]:
```
labels = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "K",
          "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U",
          "V", "W", "X", "Y"]
num_classes = len(labels)

data_dir = "Dataset_short_v3"
train_dir = os.path.join(data_dir, "train")
batch_size = 32
```

```
# X: features list, Y: class list to predict
X = []
Y = []

# read each folder of train
for label in labels:
    name = os.path.join(train_dir, label, label + ".csv")
    df = pd.read_csv(name)
    X.extend(df.values)
    newSize = len(df)
    print("Letter %s: num_data: %d" %( label, newSize))
    for i in range(0,newSize):
        Y.append(label)

# convert to np array to have shape
X = np.array(X)
Y = np.array(Y)

#one hot encoding
lb = preprocessing.LabelBinarizer()
lb.fit(labels)
Y = lb.transform(Y)
```

```
Letter A: num_data: 793
Letter B: num_data: 821
Letter C: num_data: 966
Letter D: num_data: 585
Letter E: num_data: 918
Letter F: num_data: 628
Letter G: num_data: 70
Letter H: num_data: 31
Letter I: num_data: 698
Letter K: num_data: 641
Letter L: num_data: 810
Letter M: num_data: 283
Letter N: num_data: 285
Letter O: num_data: 626
Letter P: num_data: 22
Letter Q: num_data: 11
Letter R: num_data: 544
Letter S: num_data: 546
Letter T: num_data: 260
Letter U: num_data: 595
Letter V: num_data: 768
Letter W: num_data: 818
Letter X: num_data: 347
```

```
Letter Y: num_data: 770
```

[3]:
```
sizeX = len(X)
sizeY = len(Y)
print(X.shape, Y.shape)
```

```
(12836, 42) (12836, 24)
```

[4]:
```
#split into training,val data. test_data
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.05)
X_train, X_val, Y_train, Y_val = train_test_split(X_train, Y_train, test_size=0.
↪05)
```

[5]:
```
print("train data ", X_train.shape, Y_train.shape)
print("val data ", X_val.shape, Y_val.shape)
print("test data ", X_test.shape, Y_test.shape)

#handle unbalance data
Y_train_classes = lb.inverse_transform(Y_train)
len(Y_train_classes)
from sklearn.utils import class_weight
class_weights = class_weight.compute_class_weight('balanced',
                                                  np.unique(Y_train_classes),
                                                  Y_train_classes)
class_weight_dic = dict(enumerate(class_weights))
```

```
train data  (11584, 42) (11584, 24)
val data  (610, 42) (610, 24)
test data  (642, 42) (642, 24)
```

```
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: FutureWarning: Pass classes=['A' 'B'
'C' 'D' 'E' 'F' 'G' 'H' 'I' 'K' 'L' 'M' 'N' 'O' 'P' 'Q' 'R' 'S'
 'T' 'U' 'V' 'W' 'X' 'Y'], y=['D' 'C' 'T' … 'O' 'A' 'R'] as keyword args. From
version 1.0 (renaming of 0.25) passing these as positional arguments will result
in an error
  "will result in an error", FutureWarning)
```

[6]:
```
model = Sequential()
model.add(Dense(32, activation='relu', input_shape=(42,)))
model.add(Dense(num_classes))
model.add(Activation("softmax"))
```

```
WARNING:tensorflow:From
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/keras/backend/tensorflow_backend.py:74: The name tf.get_default_graph
is deprecated. Please use tf.compat.v1.get_default_graph instead.
```

```
WARNING:tensorflow:From
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/keras/backend/tensorflow_backend.py:517: The name tf.placeholder is
deprecated. Please use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/keras/backend/tensorflow_backend.py:4138: The name tf.random_uniform is
deprecated. Please use tf.random.uniform instead.
```

[7]:
```
model.summary()
model.compile(optimizer=optimizers.SGD(lr=0.01),␣
 ↪loss="categorical_crossentropy",
                  metrics=["accuracy"])
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 32)                1376
_________________________________________________________________
dense_2 (Dense)              (None, 24)                792
_________________________________________________________________
activation_1 (Activation)    (None, 24)                0
=================================================================
Total params: 2,168
Trainable params: 2,168
Non-trainable params: 0
_________________________________________________________________
WARNING:tensorflow:From
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/keras/optimizers.py:790: The name tf.train.Optimizer is deprecated.
Please use tf.compat.v1.train.Optimizer instead.

WARNING:tensorflow:From
/home/dani/anaconda3/envs/kerasenvTFGExtractedFeatures/lib/python3.6/site-
packages/keras/backend/tensorflow_backend.py:3295: The name tf.log is
deprecated. Please use tf.math.log instead.
```

[8]:
```
checkpoint_dir = "checkpointsASLFeaturesTFGNotebook/"
checkpoint_name = (checkpoint_dir
    + "featuresTFGNotebook-{val_loss:.4f}-{val_acc:.4f}.hdf5")

if not os.path.exists(checkpoint_dir):
    os.makedirs(checkpoint_dir)
```

```
def create_callbacks():
    return [
        callbacks.EarlyStopping(
            monitor="val_acc",
            patience=15,
            verbose=1),
        callbacks.ModelCheckpoint(
            checkpoint_name,
            monitor="val_acc",
            verbose=1,
            save_best_only=True),
    ]

my_callbacks = create_callbacks()
histories = []
```

[10]:
```
#finetuning
K.set_value(model.optimizer.lr,
                K.get_value(model.optimizer.lr) / 2)

histories.append(model.fit(
    X_train,
    Y_train,
    batch_size=batch_size,
    epochs=200,
    validation_data=(X_val, Y_val),
    callbacks=my_callbacks,
    class_weight=class_weight_dic))
```

```
Train on 11584 samples, validate on 610 samples
Epoch 1/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2473 -
acc: 0.9465 - val_loss: 0.2515 - val_acc: 0.9426

Epoch 00001: val_acc did not improve from 0.94426
Epoch 2/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2451 -
acc: 0.9475 - val_loss: 0.2522 - val_acc: 0.9410

Epoch 00002: val_acc did not improve from 0.94426
Epoch 3/200
11584/11584 [==============================] - 0s 28us/step - loss: 0.2445 -
acc: 0.9476 - val_loss: 0.2472 - val_acc: 0.9393

Epoch 00003: val_acc did not improve from 0.94426
Epoch 4/200
11584/11584 [==============================] - 0s 30us/step - loss: 0.2433 -
```

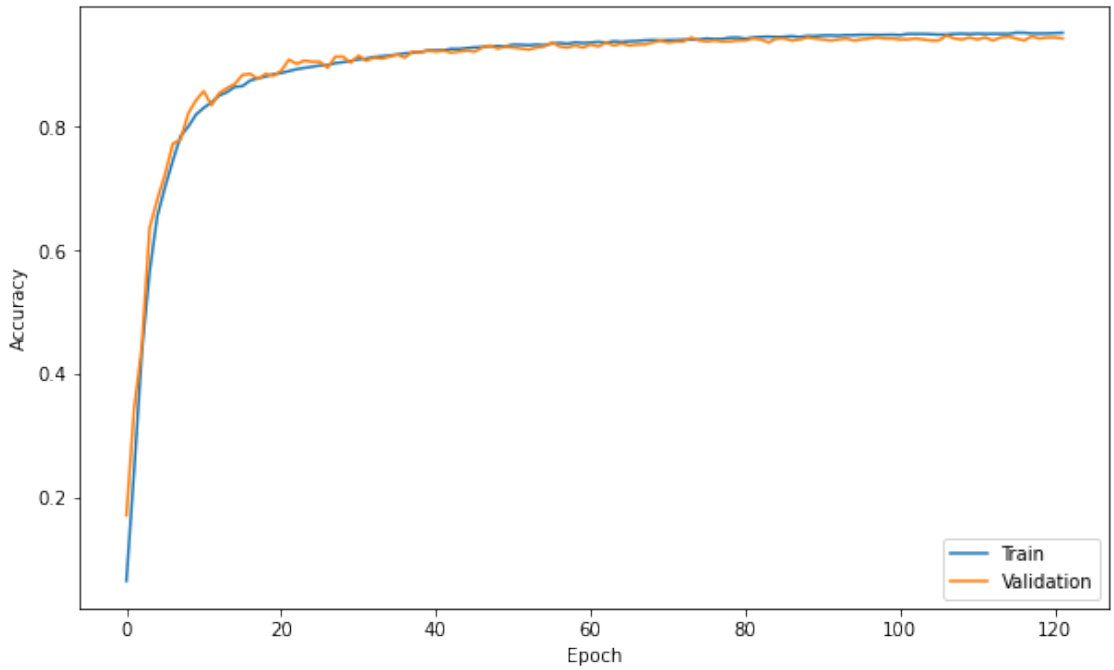
```
acc: 0.9472 - val_loss: 0.2505 - val_acc: 0.9410

Epoch 00004: val_acc did not improve from 0.94426
Epoch 5/200
11584/11584 [==============================] - 0s 28us/step - loss: 0.2427 -
acc: 0.9479 - val_loss: 0.2473 - val_acc: 0.9426

Epoch 00005: val_acc did not improve from 0.94426
Epoch 6/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2425 -
acc: 0.9480 - val_loss: 0.2471 - val_acc: 0.9393

Epoch 00006: val_acc did not improve from 0.94426
Epoch 7/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2414 -
acc: 0.9485 - val_loss: 0.2501 - val_acc: 0.9410

Epoch 00007: val_acc did not improve from 0.94426
Epoch 8/200
11584/11584 [==============================] - 0s 27us/step - loss: 0.2402 -
acc: 0.9485 - val_loss: 0.2469 - val_acc: 0.9426

Epoch 00008: val_acc did not improve from 0.94426
Epoch 9/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2404 -
acc: 0.9483 - val_loss: 0.2450 - val_acc: 0.9443

Epoch 00009: val_acc did not improve from 0.94426
Epoch 10/200
11584/11584 [==============================] - 0s 27us/step - loss: 0.2392 -
acc: 0.9485 - val_loss: 0.2455 - val_acc: 0.9426

Epoch 00010: val_acc did not improve from 0.94426
Epoch 11/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2384 -
acc: 0.9487 - val_loss: 0.2451 - val_acc: 0.9426

Epoch 00011: val_acc did not improve from 0.94426
Epoch 12/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2371 -
acc: 0.9479 - val_loss: 0.2476 - val_acc: 0.9410

Epoch 00012: val_acc did not improve from 0.94426
Epoch 13/200
11584/11584 [==============================] - 0s 27us/step - loss: 0.2373 -
acc: 0.9499 - val_loss: 0.2424 - val_acc: 0.9410

Epoch 00013: val_acc did not improve from 0.94426
```

```
Epoch 14/200
11584/11584 [==============================] - 0s 30us/step - loss: 0.2352 -
acc: 0.9499 - val_loss: 0.2418 - val_acc: 0.9426

Epoch 00014: val_acc did not improve from 0.94426
Epoch 15/200
11584/11584 [==============================] - 0s 29us/step - loss: 0.2358 -
acc: 0.9499 - val_loss: 0.2429 - val_acc: 0.9410

Epoch 00015: val_acc did not improve from 0.94426
Epoch 16/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2346 -
acc: 0.9495 - val_loss: 0.2432 - val_acc: 0.9393

Epoch 00016: val_acc did not improve from 0.94426
Epoch 17/200
11584/11584 [==============================] - 0s 28us/step - loss: 0.2334 -
acc: 0.9488 - val_loss: 0.2473 - val_acc: 0.9393

Epoch 00017: val_acc did not improve from 0.94426
Epoch 18/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2327 -
acc: 0.9492 - val_loss: 0.2425 - val_acc: 0.9475

Epoch 00018: val_acc improved from 0.94426 to 0.94754, saving model to
checkpointsASLFeaturesTFGNotebook/featuresTFGNotebook-0.2425-0.9475.hdf5
Epoch 19/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2331 -
acc: 0.9499 - val_loss: 0.2383 - val_acc: 0.9426

Epoch 00019: val_acc did not improve from 0.94754
Epoch 20/200
11584/11584 [==============================] - 0s 29us/step - loss: 0.2323 -
acc: 0.9503 - val_loss: 0.2400 - val_acc: 0.9410

Epoch 00020: val_acc did not improve from 0.94754
Epoch 21/200
11584/11584 [==============================] - 0s 27us/step - loss: 0.2310 -
acc: 0.9498 - val_loss: 0.2409 - val_acc: 0.9443

Epoch 00021: val_acc did not improve from 0.94754
Epoch 22/200
11584/11584 [==============================] - 0s 28us/step - loss: 0.2306 -
acc: 0.9504 - val_loss: 0.2370 - val_acc: 0.9410

Epoch 00022: val_acc did not improve from 0.94754
Epoch 23/200
11584/11584 [==============================] - 0s 27us/step - loss: 0.2301 -
```

```
acc: 0.9500 - val_loss: 0.2387 - val_acc: 0.9443

Epoch 00023: val_acc did not improve from 0.94754
Epoch 24/200
11584/11584 [==============================] - 0s 27us/step - loss: 0.2290 -
acc: 0.9501 - val_loss: 0.2398 - val_acc: 0.9393

Epoch 00024: val_acc did not improve from 0.94754
Epoch 25/200
11584/11584 [==============================] - 0s 29us/step - loss: 0.2281 -
acc: 0.9501 - val_loss: 0.2378 - val_acc: 0.9443

Epoch 00025: val_acc did not improve from 0.94754
Epoch 26/200
11584/11584 [==============================] - 0s 30us/step - loss: 0.2276 -
acc: 0.9499 - val_loss: 0.2332 - val_acc: 0.9459

Epoch 00026: val_acc did not improve from 0.94754
Epoch 27/200
11584/11584 [==============================] - 0s 27us/step - loss: 0.2276 -
acc: 0.9519 - val_loss: 0.2353 - val_acc: 0.9426

Epoch 00027: val_acc did not improve from 0.94754
Epoch 28/200
11584/11584 [==============================] - 0s 28us/step - loss: 0.2262 -
acc: 0.9517 - val_loss: 0.2375 - val_acc: 0.9393

Epoch 00028: val_acc did not improve from 0.94754
Epoch 29/200
11584/11584 [==============================] - 0s 30us/step - loss: 0.2262 -
acc: 0.9507 - val_loss: 0.2307 - val_acc: 0.9459

Epoch 00029: val_acc did not improve from 0.94754
Epoch 30/200
11584/11584 [==============================] - 0s 30us/step - loss: 0.2251 -
acc: 0.9505 - val_loss: 0.2344 - val_acc: 0.9426

Epoch 00030: val_acc did not improve from 0.94754
Epoch 31/200
11584/11584 [==============================] - 0s 26us/step - loss: 0.2245 -
acc: 0.9507 - val_loss: 0.2315 - val_acc: 0.9443

Epoch 00031: val_acc did not improve from 0.94754
Epoch 32/200
11584/11584 [==============================] - 0s 32us/step - loss: 0.2238 -
acc: 0.9512 - val_loss: 0.2336 - val_acc: 0.9443

Epoch 00032: val_acc did not improve from 0.94754
```

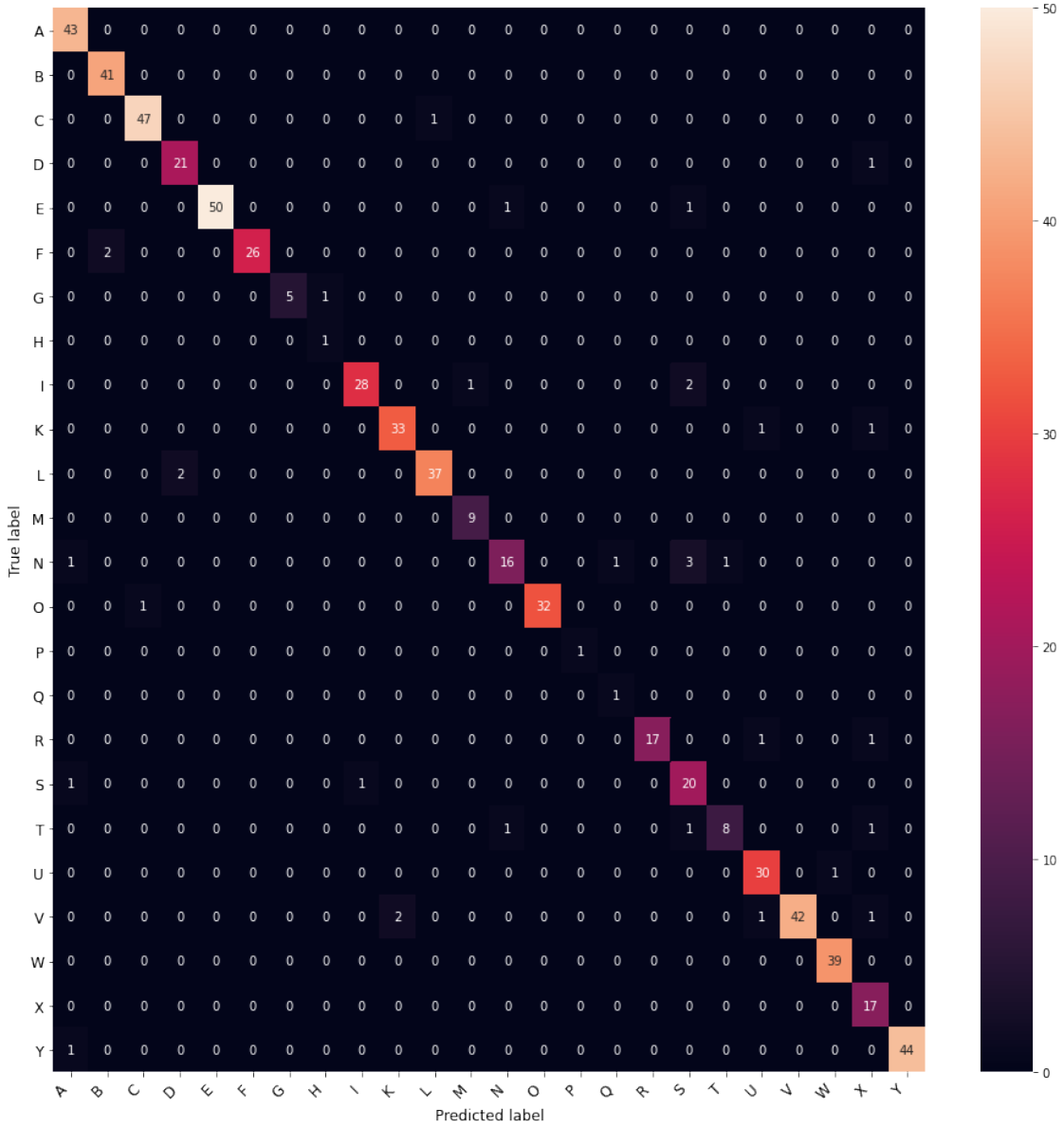
```
Epoch 33/200
11584/11584 [==============================] - 0s 29us/step - loss: 0.2226 -
acc: 0.9517 - val_loss: 0.2321 - val_acc: 0.9426

Epoch 00033: val_acc did not improve from 0.94754
Epoch 00033: early stopping
```

[11]:
```python
def combine_histories():
  history = {
      "loss": [],
      "val_loss": [],
      "acc": [],
      "val_acc": []
  }

  for h in histories:
      for k in history.keys():
          history[k] += h.history[k]
  return history

history = combine_histories()

def plot_accuracy(history):
  fig = plt.figure(figsize=(10, 6))
  plt.plot(history["acc"])
  plt.plot(history["val_acc"])
  plt.xlabel("Epoch")
  plt.ylabel("Accuracy")
  plt.legend(["Train", "Validation"])
  plt.show()

plot_accuracy(history)
```



[12]:
```
#get test data in format
probabilities = model.predict(X_test)
predicted_labels = np.argmax(probabilities, axis=-1)

target_labels_classes = lb.inverse_transform(Y_test)

#obtein index of class
le = preprocessing.LabelEncoder()
le.fit(labels)
target_labels = le.transform(target_labels_classes)

from sklearn import metrics
conf = metrics.confusion_matrix(target_labels, predicted_labels)

import seaborn as sns

def plot_confusion_matrix(conf, labels, figsize=(8, 8)):
    fig = plt.figure(figsize=figsize)
    heatmap = sns.heatmap(conf, annot=True, fmt="d")
    heatmap.xaxis.set_ticklabels(labels, rotation=45,
                                 ha="right", fontsize=12)
    heatmap.yaxis.set_ticklabels(labels, rotation=0,
                                 ha="right", fontsize=12)
    plt.xlabel("Predicted label", fontsize=12)
    plt.ylabel("True label", fontsize=12)
```

```
    plt.show()



plot_confusion_matrix(conf, labels, figsize=(16, 16))
```



[13]:
```
#metricas
print(metrics.classification_report(target_labels, predicted_labels,␣
 ↪target_names=labels))

wrong_letters = np.where(predicted_labels != target_labels)[0]
```

```
probs_max = np.max(probabilities, axis=-1)
idx = np.argsort(probs_max[wrong_letters])
idx = idx[::-1][:5]
worst_predictions = wrong_letters[idx]
worst_predictions

worst_predictions

for i in worst_predictions:
    print("was predicted as %s  %.4f" % (
            predicted_labels[i],
            probs_max[i]
    ))
```

```
              precision    recall  f1-score   support

           A       0.93      1.00      0.97        43
           B       0.95      1.00      0.98        41
           C       0.98      0.98      0.98        48
           D       0.91      0.95      0.93        22
           E       1.00      0.96      0.98        52
           F       1.00      0.93      0.96        28
           G       1.00      0.83      0.91         6
           H       0.50      1.00      0.67         1
           I       0.97      0.90      0.93        31
           K       0.94      0.94      0.94        35
           L       0.97      0.95      0.96        39
           M       0.90      1.00      0.95         9
           N       0.89      0.73      0.80        22
           O       1.00      0.97      0.98        33
           P       1.00      1.00      1.00         1
           Q       0.50      1.00      0.67         1
           R       1.00      0.89      0.94        19
           S       0.74      0.91      0.82        22
           T       0.89      0.73      0.80        11
           U       0.91      0.97      0.94        31
           V       1.00      0.91      0.95        46
           W       0.97      1.00      0.99        39
           X       0.77      1.00      0.87        17
           Y       1.00      0.98      0.99        45

    accuracy                           0.95       642
   macro avg       0.91      0.94      0.91       642
weighted avg       0.95      0.95      0.95       642

was predicted as 0  0.9520
was predicted as 3  0.8774
```

```
was predicted as 8  0.8734
was predicted as 7  0.8679
was predicted as 11  0.8001
```

[14]:
```
import coremltools
from keras.models import load_model
```

[15]:
```
best_model = load_model(checkpoint_dir + "featuresTFGNotebook-0.2425-0.9475.
↪hdf5")
```

[16]:
```
coreml_model = coremltools.converters.keras.convert(
        best_model,
        input_names="handpoint",
        output_names="labelProbability",
        predicted_feature_name="label",
        class_labels=labels)

# add metadata to the model
coreml_model.author = "Daniel Gallego Peralta"
coreml_model.license = "Public"
coreml_model.short_description = "Hand points classifier for 24 different␣
↪letters of ASL"

coreml_model.input_description["handpoint"] = "normalized coordinates for hand␣
↪points"
coreml_model.output_description["labelProbability"]= "Prediction probabilities"
coreml_model.output_description["label"]= "Class label of top prediction"

coreml_model.save("ASLHandPointTFG.mlmodel")
```

```
0 : dense_1_input, <keras.engine.input_layer.InputLayer object at
0x7f112df6b550>
1 : dense_1, <keras.layers.core.Dense object at 0x7f112df6b518>
2 : dense_1__activation__, <keras.layers.core.Activation object at
0x7f117c188cf8>
3 : dense_2, <keras.layers.core.Dense object at 0x7f112df6b940>
4 : activation_1, <keras.layers.core.Activation object at 0x7f112df72668>
```