composer-classifier

August 12, 2024

1 Predicting the Composer of a Digital Music File

1.1 AAI-511 Team 7 Final Project

Team 7: Tyler Foreman

University of San Diego, Applied Artificial Intelligence

Date: August 12, 2024

GitHub Repository: https://github.com/t4ai/music-composer-classification

[ ]:
```
!pip install pretty_midi
!pip install tensorflow_transform
```

[ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from tensorflow import keras

from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score,
 ↪recall_score, ConfusionMatrixDisplay, make_scorer

from random import sample
import os
import gc
import shutil
import pretty_midi
import librosa
import librosa.display
```

1.2 Data Organization and Exploratory Analysis

1. Extract music files for only the composers of interest - remove all others
2. Conduct EDA on the target data:

- Undersand the nature of the files and formats
- Evaluate the distribution of samples by composer/class
- Evaluate the length of the music tracks (in time)
- Identify any preparation or augmentation tasks that may be necessary

[ ]:
```python
# mount google drive
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[ ]:
```python
# setup target data locations
root_data_path = '/content/drive/MyDrive/USD/datasets/composers_music'
target_data_path = '/content/drive/MyDrive/USD/datasets/composers_music/target'
```

[ ]:
```python
# setup data prep parameters
SETUP_MODE = False
SAMPLE_FREQUENCY = 20
NUM_PIANO_KEYS = 128

# helper function for moving and flattening directories by composer
def move_and_flatten(composer_name):
  # setup destination
  target_path = target_data_path + '/' + composer_name
  os.makedirs(target_path, exist_ok=True)

  # get source dir and subdirs
  composer_path = root_data_path + '/midiclassics/' + composer_name
  composer_dirs = [x[0] for x in os.walk(composer_path)]

  # traverse directories
  num_files = 0
  for dir in composer_dirs:
    for filename in os.listdir(dir):
      if os.path.isfile(os.path.join(dir, filename)):
        shutil.copyfile(dir + '/' + filename, target_path + '/' + filename)
        num_files += 1
  print(f'Moved {num_files} files for {composer_name}')

# helper function for loading piano rolls for a composer
def load_piano_rolls(composer_name, frequency):
  piano_rolls = []
  target_path = target_data_path + '/' + composer_name
  for filename in os.listdir(target_path):
    if filename.lower().endswith('.mid'):
      try:
        midi = pretty_midi.PrettyMIDI(target_path + '/' + filename)
```

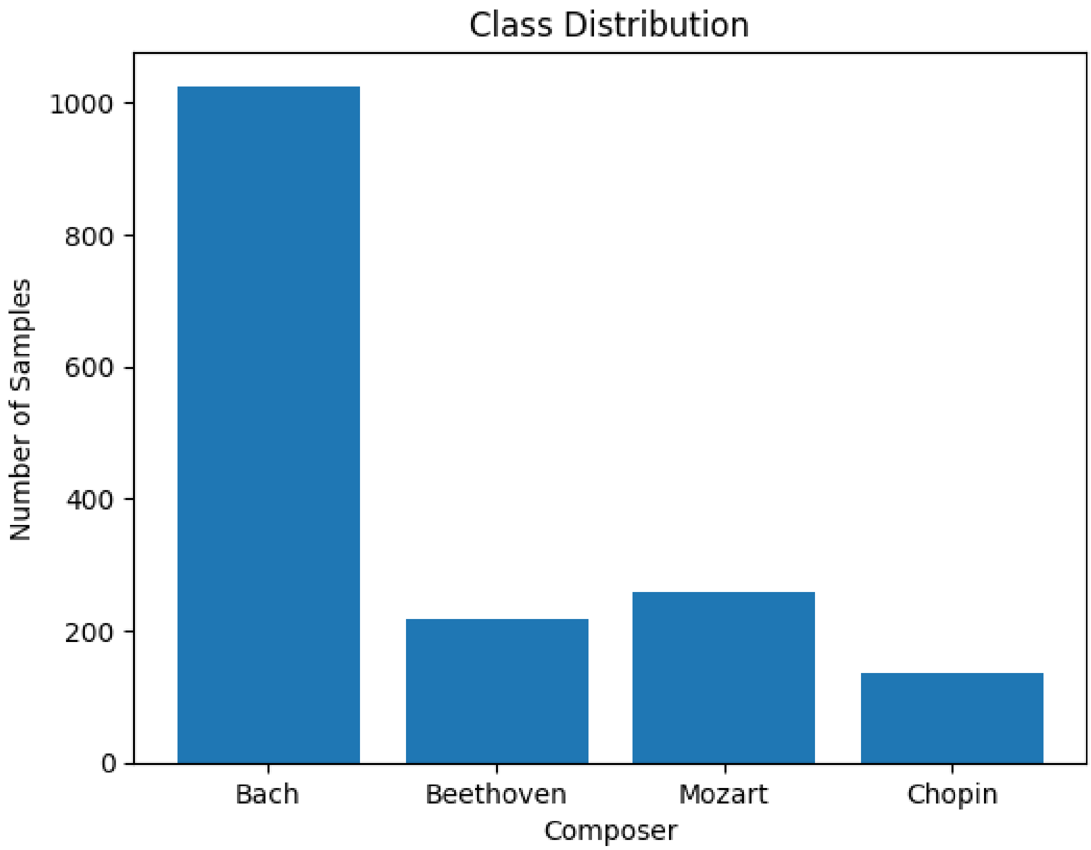
```
        midi.remove_invalid_notes()
        piano_rolls.append(midi.get_piano_roll(fs=frequency))
      except Exception as e:
        print(f'Error reading {filename}: {e}')
  return piano_rolls
```

[ ]:
```
# extract only the files from our taget 4 composers: Bach, Beethoven, Mozart,␣
↪Chopin and flatten the folder structure
if SETUP_MODE==True:

  os.makedirs(target_data_path, exist_ok=True)

  move_and_flatten('Bach')
  move_and_flatten('Beethoven')
  move_and_flatten('Mozart')
  move_and_flatten('Chopin')
```

[ ]:
```
# print total number of music files
print(f'Total number of music files: {sum([len(files) for r, d, files in os.
↪walk(target_data_path)])}')
```

```
Total number of music files: 1637
```

[ ]:
```
# visualize class balances
class_samples = {"Bach": 1025, "Beethoven": 219, "Mozart": 257, "Chopin": 136}
plt.bar(class_samples.keys(), class_samples.values())
plt.xlabel('Composer')
plt.ylabel('Number of Samples')
plt.title('Class Distribution')
plt.show()
```



Feature Extraction We will use the prettyMIDI library to extract features from the raw midi files. For this project, the focus will be on extracting the "piano roll" sequences for each track. The piano roll essentially converts all the instrument sounds in the track into sequnce of piano keys and intensity. The output is a 128 dimension vector, with each dimension representing a key and its value representing the intesity of the key at the time step.

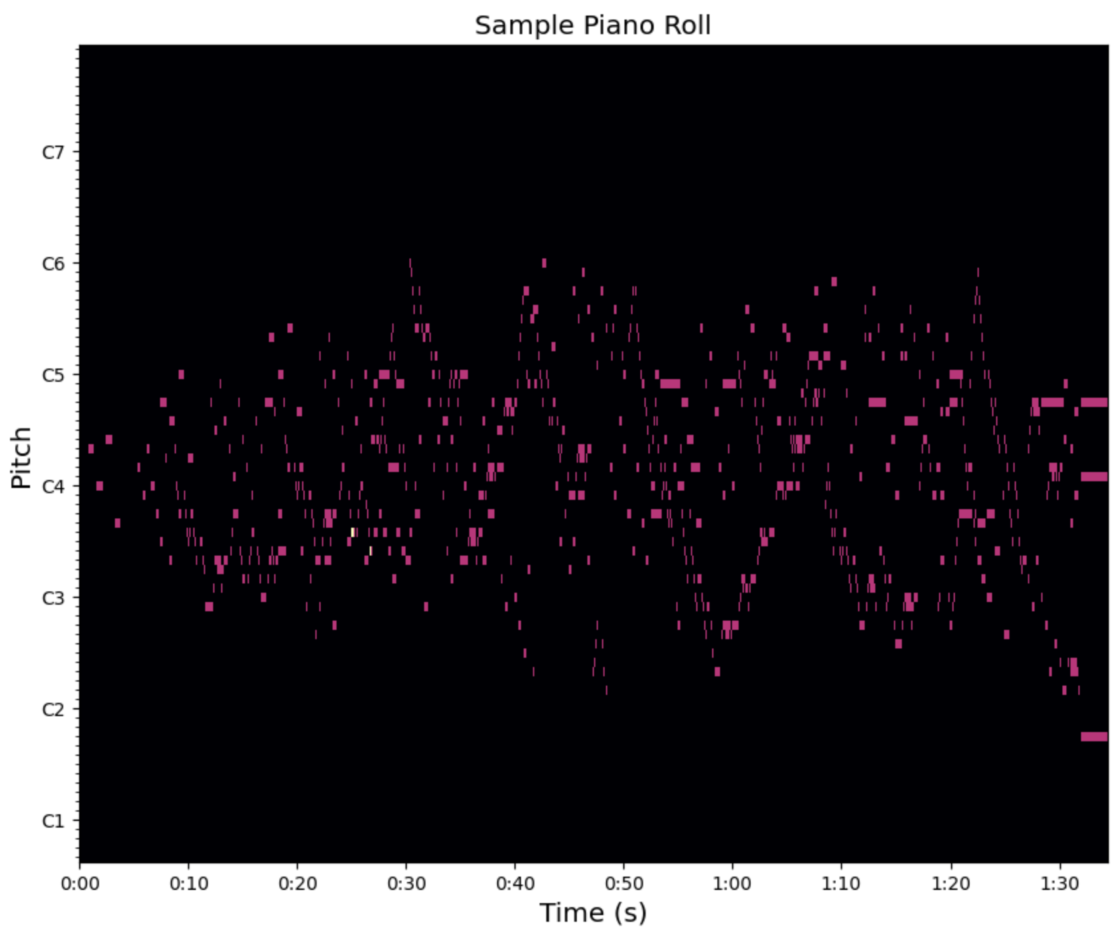
[ ]:
```
# get the distribution of sequence lengths of the pieces (in time)
bach_data = load_piano_rolls('Bach', SAMPLE_FREQUENCY)
beethoven_data = load_piano_rolls('Beethoven', SAMPLE_FREQUENCY)
mozart_data = load_piano_rolls('Mozart', SAMPLE_FREQUENCY)
chopin_data = load_piano_rolls('Chopin', SAMPLE_FREQUENCY)
```

```
/usr/local/lib/python3.10/dist-packages/pretty_midi/pretty_midi.py:100:
RuntimeWarning: Tempo, Key or Time signature change events found on non-zero
tracks.  This is not a valid type 0 or type 1 MIDI file.  Tempo, Key or Time
Signature may be wrong.
  warnings.warn(
```

```
Error reading Anhang 14-3.mid: Could not decode key with 3 flats and mode 255
Error reading K281 Piano Sonata n03 3mov.mid: Could not decode key with 2 flats
and mode 2
```

[ ]:
```
# visualize a piano roll
start_pitch = 20
end_pitch = 108
pr = bach_data[0]

fig = plt.figure(figsize=(10, 8))
librosa.display.specshow(pr[start_pitch:end_pitch],
                         hop_length=1, sr=fs, x_axis='time', y_axis='cqt_note',
                         fmin=pretty_midi.note_number_to_hz(start_pitch))
plt.title(f"Sample Piano Roll", fontsize="x-large")
plt.xlabel("Time (s)", fontsize="x-large")
plt.ylabel("Pitch", fontsize="x-large")
plt.show()
```

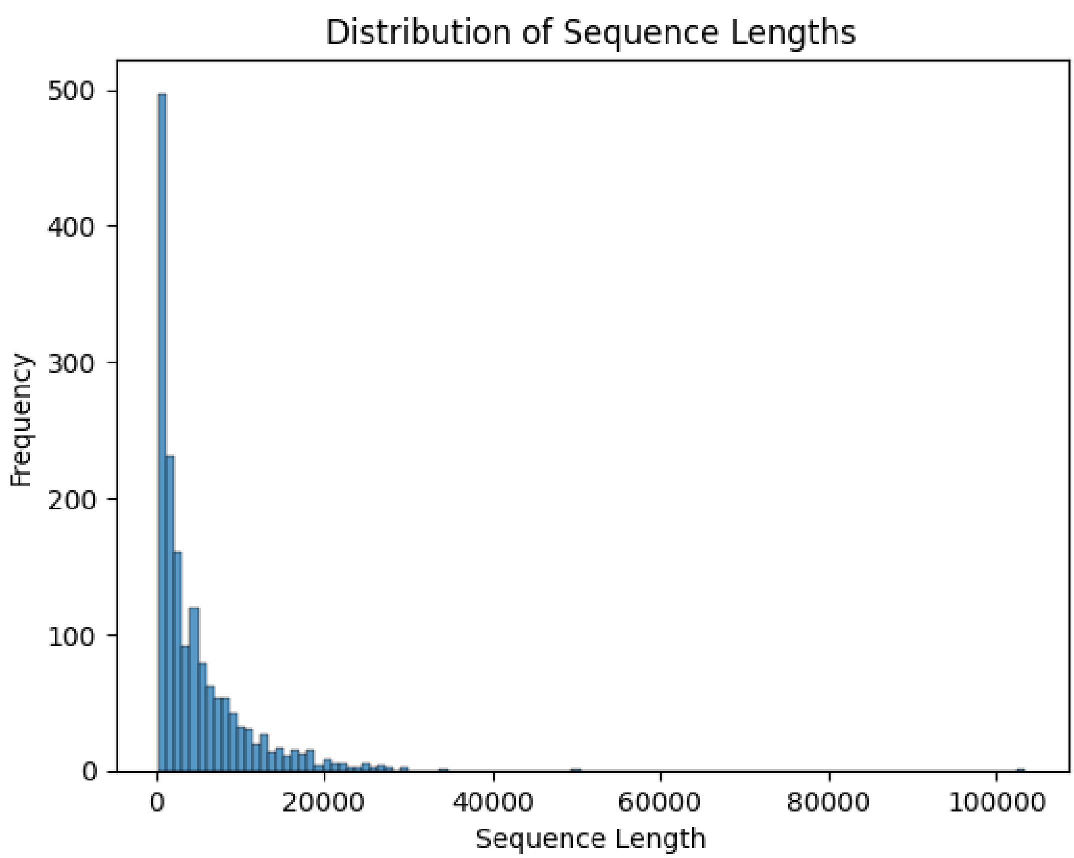
```
<ipython-input-31-8eb8a0f945cd>:7: UserWarning: Frequency axis exceeds Nyquist.
Did you remember to set all spectrogram parameters in specshow?
  librosa.display.specshow(pr[start_pitch:end_pitch],
```



```
[ ]: # build length distribution
length_distributions = []
def append_lengths(data):
  for i in range(len(data)):
    length_distributions.append(data[i].shape[1])
```

```
[ ]: # concatenate all classes
append_lengths(bach_data)
append_lengths(beethoven_data)
append_lengths(mozart_data)
append_lengths(chopin_data)
```

```
[ ]: # plot the distribution
sns.histplot(length_distributions)
plt.xlabel('Sequence Length')
plt.ylabel('Frequency')
plt.title('Distribution of Sequence Lengths')
plt.show()
```



[ ]:
```
# get descriptive statistics for sequence lengths
stats_df = pd.DataFrame(length_distributions)
stats_df.describe()
```

[ ]:

	0
count	1628.000000
mean	4838.340909
std	5900.262705
min	350.000000
25%	1019.750000
50%	2647.000000
75%	6518.250000
max	103362.000000

Analysis There is a significant class imbalance, where Bach pieces far outweigh the number of pieces by the other composers. Some balancing will be required - likely starting with using a subset of the Bach pieces.

There is also a wide range of sequence values. Some preparation tasks will likely be required to normalize these to a standard sequence length for the model. We can use the descriptive statistics to determine this - possibly breaking the larger tracks into multiple samples, each with a smaller sequence length.

1.3 Data Preparation

1. Fix the class imbalance issue by downsampling the over-represented class (Bach)
2. Normalize the data using scaler
3. Process the tracks into smaller, normalized sequence lengths
4. Format the dataset into samples and labels suitable for model input
5. Split the result into train/test/val

[ ]:
```
# set a normalized sequence length to 10 s worth of samples
NORM_SEQUENCE_LENGTH = SAMPLE_FREQUENCY * 10
OVERSAMPLE_STRIDE = 10

# helper function to pad sequences not quite long enough
def zero_pad(seq, x, missing):
        right = np.zeros((x, missing))
        return np.hstack((seq,right))

# helper function to process track into sequences of normalized length
def process_track_to_sequences(track):
  x,y = track.shape
  #print(f"track shape:", track.shape)
  sequences = []
  if(y < NORM_SEQUENCE_LENGTH):
    sequence = zero_pad(track, x, NORM_SEQUENCE_LENGTH-y)
    sequences.append(sequence)
```

```
  else:
    num_sequences = int(np.ceil(y / NORM_SEQUENCE_LENGTH))
    for i in range(num_sequences):
      start = i * NORM_SEQUENCE_LENGTH
      if start + NORM_SEQUENCE_LENGTH > y:
        sequence = track[:,start:y]
        zp = zero_pad(sequence, x, (start + NORM_SEQUENCE_LENGTH - y))
        sequences.append(zp)
      else:
        end = start + NORM_SEQUENCE_LENGTH
        sequence = track[:,start:end]
        sequences.append(sequence)

  return sequences

# helper function to process track into sequences of normalized length
def process_track_to_sequences_oversample(track):
  x,y = track.shape
  sequences = []
  if(y < NORM_SEQUENCE_LENGTH):
    sequence = zero_pad(track, x, NORM_SEQUENCE_LENGTH-y)
    sequences.append(sequence)
  else:
    start = 0
    end = 0
    while(end < y):
      end = start + NORM_SEQUENCE_LENGTH
      if end > y:
        sequence = track[:,start:y]
        zp = zero_pad(sequence, x, (start + NORM_SEQUENCE_LENGTH - y))
        sequences.append(zp)
      else:
        sequence = track[:,start:end]
        sequences.append(sequence)
      start += OVERSAMPLE_STRIDE

  return sequences

# helper function to process all tracks for a composer into sequences
def process_composer_sequences(composer_data, do_oversample):
  composer_sequences = []
  for i in range(len(composer_data)):
    if do_oversample:
      sequences = process_track_to_sequences_oversample(composer_data[i])
    else:
      sequences = process_track_to_sequences(composer_data[i])
    for seq in sequences:
```

```
      composer_sequences.append(seq)
  return composer_sequences

# helper function to transpose sequences
def transpose_sequences(sequences):
  for i in range(len(sequences)):
    sequences[i] = np.transpose(sequences[i])
  return sequences
```

[ ]:
```
# process data into sequences for each composer
bach_sequences = process_composer_sequences(bach_data, True)
beethoven_sequences = process_composer_sequences(beethoven_data, True)
mozart_sequences = process_composer_sequences(mozart_data, True)
chopin_sequences = process_composer_sequences(chopin_data, True)
```

[ ]:
```
# transpose the sequences
bach_sequences = transpose_sequences(bach_sequences)
beethoven_sequences = transpose_sequences(beethoven_sequences)
mozart_sequences = transpose_sequences(mozart_sequences)
chopin_sequences = transpose_sequences(chopin_sequences)
```

[ ]:
```
# display sequence lenghths:
print(f'Bach: {len(bach_sequences)}')
print(f'Beethoven: {len(beethoven_sequences)}')
print(f'Mozart: {len(mozart_sequences)}')
print(f'Chopin: {len(chopin_sequences)}')
```

```
Bach: 298248
Beethoven: 201956
Mozart: 199715
Chopin: 57416
```

[ ]:
```
#randomly downsample Bach, Beethoven and Mozart sequences to better balance␣
↪with lower Chopin samples
bach_sequences_sampled = sample(bach_sequences, 12000)
beethoven_sequences_sampled = sample(beethoven_sequences, 15000)
mozart_sequences_sampled = sample(mozart_sequences, 12000)
chopin_sequences_sampled = sample(chopin_sequences, 15000)
```

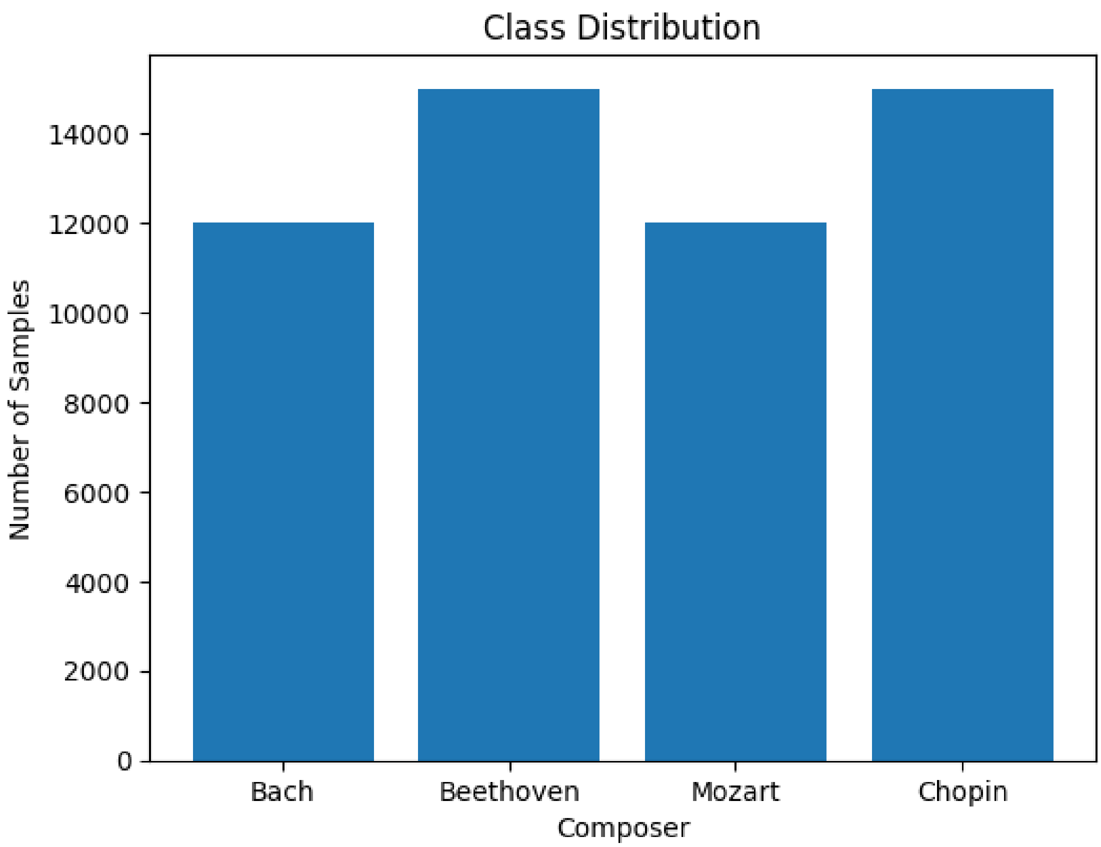
[ ]:
```
# display sequence lenghths:
print(f'Bach: {len(bach_sequences_sampled)}')
print(f'Beethoven: {len(beethoven_sequences_sampled)}')
print(f'Mozart: {len(mozart_sequences_sampled)}')
print(f'Chopin: {len(chopin_sequences_sampled)}')
```

```
Bach: 12000
Beethoven: 15000
Mozart: 12000
```

```
Chopin: 15000
```

[ ]:

```
# visualize class balances
class_samples = {"Bach": len(bach_sequences_sampled), "Beethoven":␣
 ↪len(beethoven_sequences_sampled), "Mozart": len(mozart_sequences_sampled),␣
 ↪"Chopin": len(chopin_sequences_sampled)}
plt.bar(class_samples.keys(), class_samples.values())
plt.xlabel('Composer')
plt.ylabel('Number of Samples')
plt.title('Class Distribution')
plt.show()
```



[ ]:

```
# create labels
bach_labels = ['bach'] * len(bach_sequences_sampled)
beethoven_labels = ['beethoven'] * len(beethoven_sequences_sampled)
mozart_labels = ['mozart'] * len(mozart_sequences_sampled)
chopin_labels = ['chopin'] * len(chopin_sequences_sampled)
```

[ ]:

```
# free up RAM
del bach_sequences
```

```
del beethoven_sequences
del mozart_sequences
del chopin_sequences
del bach_data
del beethoven_data
del mozart_data
del chopin_data
gc.collect()
```

[ ]: 0

[ ]:
```
# next stack the sequences into single list
X = np.concatenate((bach_sequences_sampled, beethoven_sequences_sampled,␣
 ↪mozart_sequences_sampled, chopin_sequences_sampled))
y_raw = np.concatenate((bach_labels, beethoven_labels, mozart_labels,␣
 ↪chopin_labels))
```

[ ]:
```
# free up RAM
del bach_sequences_sampled
del beethoven_sequences_sampled
del mozart_sequences_sampled
del chopin_sequences_sampled
gc.collect()
```

[ ]: 0

[ ]:
```
# shuffle the dataset
indices = np.arange(X.shape[0])
np.random.seed(23)
np.random.shuffle(indices, )

X = X[indices]
y_raw = y_raw[indices]
```

[ ]:
```
# label encode the labels
ohe = OneHotEncoder()
y = ohe.fit_transform(y_raw.reshape(-1, 1))
y = y.toarray()
y
```

[ ]:
```
array([[0., 0., 0., 1.],
       [0., 1., 0., 0.],
       [0., 1., 0., 0.],
       …,
       [0., 1., 0., 0.],
       [1., 0., 0., 0.],
       [0., 0., 1., 0.]])
```

[ ]:
```
ohe.categories_
```

[ ]: [array(['bach', 'beethoven', 'chopin', 'mozart'], dtype='<U9')]

[ ]:
```
# split into train/test/val
TRAIN_SPLIT = 0.8
TEST_VAL_SPLIT = 0.1
TOTAL_LEN = len(X)

# train data
X_train = X[:int(TOTAL_LEN * TRAIN_SPLIT)].copy()
y_train = y[:int(TOTAL_LEN * TRAIN_SPLIT)].copy()

# val data
X_val = X[int(TOTAL_LEN * TRAIN_SPLIT):int(TOTAL_LEN * (TRAIN_SPLIT +␣
  ↪TEST_VAL_SPLIT))].copy()
y_val = y[int(TOTAL_LEN * TRAIN_SPLIT):int(TOTAL_LEN * (TRAIN_SPLIT +␣
  ↪TEST_VAL_SPLIT))].copy()

# test data
X_test = X[int(TOTAL_LEN * (TRAIN_SPLIT + TEST_VAL_SPLIT)):].copy()
y_test = y[int(TOTAL_LEN * (TRAIN_SPLIT + TEST_VAL_SPLIT)):].copy()
```

[ ]:
```
#helper function to get min/max range from train data tensors
def normalize_dataset(dataset, min, max):
  for i in range(len(dataset)):
    sample = dataset[i]
    sample = [((x - min) / (max - min)) for x in sample]
    dataset[i] = sample
  return dataset
```

[ ]:
```
# set min-max scale ranges
scale_max = np.max(X_train)
scale_min = np.min(X_train)
print(scale_min, scale_max)
```

```
0.0 1143.0
```

[ ]:
```
# normalize the datasets [normalize based on training data]
#X_train_norm = normalize_dataset(X_train.copy(), scale_min, scale_max)
#X_val_norm = normalize_dataset(X_val.copy(), scale_min, scale_max)
#X_test_norm = normalize_dataset(X_test.copy(), scale_min, scale_max)
```

[ ]:
```
# free up RAM
del X
del y
del y_raw
gc.collect()
```

```
[ ]: 0
```

```
[ ]: # write final dataset for re-use
np.savez_compressed(root_data_path + '/prepared/train.npy', a=X_train,␣
 ↪b=y_train)
np.savez_compressed(root_data_path + '/prepared/test.npy', a=X_test, b=y_test)
np.savez_compressed(root_data_path + '/prepared/val.npy', a=X_val, b=y_val)
```

```
[ ]: # helper function to re-load data
def load_prepared_data():

  # load train
  train_loaded = np.load(root_data_path + '/prepared/train.npy.npz')
  X_train_loaded = train_loaded['a']
  y_train_loaded = train_loaded['b']

  # load val
  val_loaded = np.load(root_data_path + '/prepared/val.npy.npz')
  X_val_loaded = val_loaded['a']
  y_val_loaded = val_loaded['b']

  # load test
  test_loaded = np.load(root_data_path + '/prepared/test.npy.npz')
  X_test_loaded = test_loaded['a']
  y_test_loaded = test_loaded['b']


  return X_train_loaded, y_train_loaded, X_val_loaded, y_val_loaded,␣
 ↪X_test_loaded, y_test_loaded
```

```
[ ]: X_train.shape
```

```
[ ]: (43200, 200, 128)
```

```
[ ]: # load dataset
X_train, y_train, X_val, y_val, X_test, y_test = load_prepared_data()
```

1.4 Model Definition and Experimentation

With the dataset prepared, we will now experiment with different model architectures and configurations. For this project, we will evaluate two primary architectures: LSTM and CNN. For each architecture, a baseline model will be defined and then additional experiments will be conducted, each changing different aspects of the model configuration including layer depth, number of nodes per layer and hyperparameters. The models will be evaluated against the validation set and measured using metrics Categorical Accuracy, Precision, Recall and F1 score.

```
[ ]: # global training parameters
NUM_EPOCHS = 75
```

```
BATCH_SIZE = 32
LEARNING_RATE = 0.001
NUM_COMPOSERS = 4
```

1.4.1 LSTM Models

For our set of models, we will define and train and LSTM models to process our sequences and perform a classification task to predict the appropriate composer. Some experimenation and fine tuning will be conducted to find an optimal model definition.

1. Define baseline LSTM model with classification output layer. This will be used to validate our processed data, validate classification task and set baseline performance.
2. Train model on our training set
3. Evaluate performance of the model using Accuracy, Precision/Recall, F1
4. Tune hyperparameters and model architecture

Baseline LSTM This is a simple LSTM with a single hidden layer with 256 units. There is also only a single fully connected layer with 64 units. No dropout or regularization techniques are applied.

[ ]:
```
# define a baseline LSTM
lstm_base = tf.keras.models.Sequential([

    # input our sequence tensors
    tf.keras.layers.Input(shape=(NORM_SEQUENCE_LENGTH, NUM_PIANO_KEYS)),
    tf.keras.layers.Normalization(axis=None),
    tf.keras.layers.LSTM(256, return_sequences=False, dropout=0.2),

    # classification head
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(units = NUM_COMPOSERS, activation='softmax')
])

# Compile the model
lstm_base.compile(optimizer='adam', loss='categorical_crossentropy',␣
 ↪metrics=[keras.metrics.CategoricalAccuracy(), keras.metrics.Precision(),␣
 ↪keras.metrics.Recall(), keras.metrics.F1Score()])
```

[ ]:
```
# Train the model
history_lstm = lstm_base.fit(X_train, y_train, validation_data=(X_val,y_val),␣
 ↪epochs=NUM_EPOCHS, batch_size=BATCH_SIZE)
```

```
Epoch 1/100
1000/1000 [==============================] - 18s 13ms/step - loss: 1.1398 -
categorical_accuracy: 0.4942 - precision: 0.6264 - recall: 0.2708 - f1_score:
0.4892 - val_loss: 0.9461 - val_categorical_accuracy: 0.5920 - val_precision:
0.6872 - val_recall: 0.4593 - val_f1_score: 0.5855
Epoch 2/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.8743 -
```

```
categorical_accuracy: 0.6299 - precision: 0.6980 - recall: 0.5195 - f1_score:
0.6244 - val_loss: 0.8611 - val_categorical_accuracy: 0.6398 - val_precision:
0.6945 - val_recall: 0.5477 - val_f1_score: 0.6251
Epoch 3/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.7755 -
categorical_accuracy: 0.6785 - precision: 0.7295 - recall: 0.5979 - f1_score:
0.6738 - val_loss: 0.8175 - val_categorical_accuracy: 0.6545 - val_precision:
0.6998 - val_recall: 0.5867 - val_f1_score: 0.6449
Epoch 4/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.7189 -
categorical_accuracy: 0.7000 - precision: 0.7461 - recall: 0.6371 - f1_score:
0.6962 - val_loss: 0.7751 - val_categorical_accuracy: 0.6708 - val_precision:
0.7192 - val_recall: 0.6018 - val_f1_score: 0.6680
Epoch 5/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.6730 -
categorical_accuracy: 0.7223 - precision: 0.7637 - recall: 0.6698 - f1_score:
0.7193 - val_loss: 0.7662 - val_categorical_accuracy: 0.6790 - val_precision:
0.7148 - val_recall: 0.6223 - val_f1_score: 0.6722
Epoch 6/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.6466 -
categorical_accuracy: 0.7360 - precision: 0.7713 - recall: 0.6856 - f1_score:
0.7333 - val_loss: 0.7570 - val_categorical_accuracy: 0.6842 - val_precision:
0.7187 - val_recall: 0.6317 - val_f1_score: 0.6825
Epoch 7/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.6195 -
categorical_accuracy: 0.7474 - precision: 0.7805 - recall: 0.7053 - f1_score:
0.7448 - val_loss: 0.7330 - val_categorical_accuracy: 0.6957 - val_precision:
0.7250 - val_recall: 0.6497 - val_f1_score: 0.6941
Epoch 8/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.5894 -
categorical_accuracy: 0.7590 - precision: 0.7903 - recall: 0.7224 - f1_score:
0.7568 - val_loss: 0.7335 - val_categorical_accuracy: 0.7017 - val_precision:
0.7297 - val_recall: 0.6628 - val_f1_score: 0.6963
Epoch 9/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.5597 -
categorical_accuracy: 0.7738 - precision: 0.8019 - recall: 0.7391 - f1_score:
0.7719 - val_loss: 0.7129 - val_categorical_accuracy: 0.7085 - val_precision:
0.7351 - val_recall: 0.6700 - val_f1_score: 0.7049
Epoch 10/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.5611 -
categorical_accuracy: 0.7719 - precision: 0.8007 - recall: 0.7377 - f1_score:
0.7701 - val_loss: 0.6945 - val_categorical_accuracy: 0.7157 - val_precision:
0.7472 - val_recall: 0.6790 - val_f1_score: 0.7089
Epoch 11/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.5397 -
categorical_accuracy: 0.7854 - precision: 0.8117 - recall: 0.7511 - f1_score:
0.7839 - val_loss: 0.6716 - val_categorical_accuracy: 0.7325 - val_precision:
0.7634 - val_recall: 0.6967 - val_f1_score: 0.7288
```

```
Epoch 12/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.5271 -
categorical_accuracy: 0.7903 - precision: 0.8149 - recall: 0.7595 - f1_score:
0.7889 - val_loss: 0.6731 - val_categorical_accuracy: 0.7207 - val_precision:
0.7507 - val_recall: 0.6842 - val_f1_score: 0.7175
Epoch 13/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.5154 -
categorical_accuracy: 0.7937 - precision: 0.8182 - recall: 0.7629 - f1_score:
0.7924 - val_loss: 0.6891 - val_categorical_accuracy: 0.7305 - val_precision:
0.7585 - val_recall: 0.6965 - val_f1_score: 0.7252
Epoch 14/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.5040 -
categorical_accuracy: 0.7976 - precision: 0.8220 - recall: 0.7685 - f1_score:
0.7964 - val_loss: 0.6981 - val_categorical_accuracy: 0.7237 - val_precision:
0.7558 - val_recall: 0.6917 - val_f1_score: 0.7199
Epoch 15/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4981 -
categorical_accuracy: 0.8008 - precision: 0.8245 - recall: 0.7728 - f1_score:
0.7996 - val_loss: 0.7006 - val_categorical_accuracy: 0.7195 - val_precision:
0.7460 - val_recall: 0.6940 - val_f1_score: 0.7183
Epoch 16/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4849 -
categorical_accuracy: 0.8075 - precision: 0.8279 - recall: 0.7814 - f1_score:
0.8064 - val_loss: 0.6894 - val_categorical_accuracy: 0.7305 - val_precision:
0.7561 - val_recall: 0.7005 - val_f1_score: 0.7260
Epoch 17/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4769 -
categorical_accuracy: 0.8084 - precision: 0.8326 - recall: 0.7839 - f1_score:
0.8074 - val_loss: 0.7065 - val_categorical_accuracy: 0.7333 - val_precision:
0.7533 - val_recall: 0.7067 - val_f1_score: 0.7297
Epoch 18/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4718 -
categorical_accuracy: 0.8131 - precision: 0.8343 - recall: 0.7886 - f1_score:
0.8120 - val_loss: 0.6738 - val_categorical_accuracy: 0.7390 - val_precision:
0.7597 - val_recall: 0.7153 - val_f1_score: 0.7386
Epoch 19/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4672 -
categorical_accuracy: 0.8144 - precision: 0.8372 - recall: 0.7922 - f1_score:
0.8133 - val_loss: 0.6627 - val_categorical_accuracy: 0.7450 - val_precision:
0.7649 - val_recall: 0.7230 - val_f1_score: 0.7419
Epoch 20/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4546 -
categorical_accuracy: 0.8211 - precision: 0.8418 - recall: 0.7981 - f1_score:
0.8205 - val_loss: 0.6711 - val_categorical_accuracy: 0.7380 - val_precision:
0.7582 - val_recall: 0.7147 - val_f1_score: 0.7335
Epoch 21/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4564 -
categorical_accuracy: 0.8197 - precision: 0.8417 - recall: 0.7960 - f1_score:
```

```
0.8190 - val_loss: 0.6747 - val_categorical_accuracy: 0.7400 - val_precision:
0.7615 - val_recall: 0.7153 - val_f1_score: 0.7361
Epoch 22/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4433 -
categorical_accuracy: 0.8234 - precision: 0.8451 - recall: 0.8015 - f1_score:
0.8225 - val_loss: 0.6620 - val_categorical_accuracy: 0.7460 - val_precision:
0.7679 - val_recall: 0.7220 - val_f1_score: 0.7402
Epoch 23/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4469 -
categorical_accuracy: 0.8253 - precision: 0.8455 - recall: 0.8028 - f1_score:
0.8244 - val_loss: 0.7034 - val_categorical_accuracy: 0.7310 - val_precision:
0.7517 - val_recall: 0.7130 - val_f1_score: 0.7271
Epoch 24/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4404 -
categorical_accuracy: 0.8265 - precision: 0.8468 - recall: 0.8056 - f1_score:
0.8256 - val_loss: 0.6685 - val_categorical_accuracy: 0.7393 - val_precision:
0.7632 - val_recall: 0.7145 - val_f1_score: 0.7381
Epoch 25/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4299 -
categorical_accuracy: 0.8312 - precision: 0.8513 - recall: 0.8108 - f1_score:
0.8306 - val_loss: 0.6656 - val_categorical_accuracy: 0.7450 - val_precision:
0.7642 - val_recall: 0.7210 - val_f1_score: 0.7410
Epoch 26/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4261 -
categorical_accuracy: 0.8328 - precision: 0.8514 - recall: 0.8135 - f1_score:
0.8321 - val_loss: 0.6569 - val_categorical_accuracy: 0.7445 - val_precision:
0.7643 - val_recall: 0.7270 - val_f1_score: 0.7440
Epoch 27/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4192 -
categorical_accuracy: 0.8371 - precision: 0.8550 - recall: 0.8173 - f1_score:
0.8365 - val_loss: 0.6993 - val_categorical_accuracy: 0.7355 - val_precision:
0.7576 - val_recall: 0.7143 - val_f1_score: 0.7330
Epoch 28/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4151 -
categorical_accuracy: 0.8349 - precision: 0.8526 - recall: 0.8153 - f1_score:
0.8342 - val_loss: 0.6823 - val_categorical_accuracy: 0.7455 - val_precision:
0.7650 - val_recall: 0.7283 - val_f1_score: 0.7428
Epoch 29/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4235 -
categorical_accuracy: 0.8324 - precision: 0.8517 - recall: 0.8115 - f1_score:
0.8318 - val_loss: 0.6859 - val_categorical_accuracy: 0.7393 - val_precision:
0.7577 - val_recall: 0.7200 - val_f1_score: 0.7360
Epoch 30/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4141 -
categorical_accuracy: 0.8373 - precision: 0.8550 - recall: 0.8191 - f1_score:
0.8366 - val_loss: 0.6671 - val_categorical_accuracy: 0.7502 - val_precision:
0.7722 - val_recall: 0.7287 - val_f1_score: 0.7471
Epoch 31/100
```

```
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4102 -
categorical_accuracy: 0.8387 - precision: 0.8566 - recall: 0.8196 - f1_score:
0.8381 - val_loss: 0.6706 - val_categorical_accuracy: 0.7515 - val_precision:
0.7695 - val_recall: 0.7345 - val_f1_score: 0.7472
Epoch 32/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4062 -
categorical_accuracy: 0.8410 - precision: 0.8567 - recall: 0.8217 - f1_score:
0.8402 - val_loss: 0.6456 - val_categorical_accuracy: 0.7552 - val_precision:
0.7761 - val_recall: 0.7375 - val_f1_score: 0.7517
Epoch 33/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4060 -
categorical_accuracy: 0.8415 - precision: 0.8585 - recall: 0.8232 - f1_score:
0.8409 - val_loss: 0.6676 - val_categorical_accuracy: 0.7490 - val_precision:
0.7690 - val_recall: 0.7250 - val_f1_score: 0.7445
Epoch 34/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4022 -
categorical_accuracy: 0.8427 - precision: 0.8600 - recall: 0.8243 - f1_score:
0.8421 - val_loss: 0.6775 - val_categorical_accuracy: 0.7405 - val_precision:
0.7602 - val_recall: 0.7203 - val_f1_score: 0.7369
Epoch 35/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.4032 -
categorical_accuracy: 0.8435 - precision: 0.8611 - recall: 0.8256 - f1_score:
0.8429 - val_loss: 0.6448 - val_categorical_accuracy: 0.7492 - val_precision:
0.7690 - val_recall: 0.7300 - val_f1_score: 0.7458
Epoch 36/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3915 -
categorical_accuracy: 0.8432 - precision: 0.8613 - recall: 0.8256 - f1_score:
0.8426 - val_loss: 0.6665 - val_categorical_accuracy: 0.7542 - val_precision:
0.7703 - val_recall: 0.7352 - val_f1_score: 0.7501
Epoch 37/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3936 -
categorical_accuracy: 0.8459 - precision: 0.8620 - recall: 0.8300 - f1_score:
0.8454 - val_loss: 0.6814 - val_categorical_accuracy: 0.7450 - val_precision:
0.7625 - val_recall: 0.7255 - val_f1_score: 0.7443
Epoch 38/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3905 -
categorical_accuracy: 0.8467 - precision: 0.8628 - recall: 0.8287 - f1_score:
0.8462 - val_loss: 0.6885 - val_categorical_accuracy: 0.7365 - val_precision:
0.7542 - val_recall: 0.7172 - val_f1_score: 0.7343
Epoch 39/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3969 -
categorical_accuracy: 0.8455 - precision: 0.8612 - recall: 0.8270 - f1_score:
0.8449 - val_loss: 0.6801 - val_categorical_accuracy: 0.7508 - val_precision:
0.7687 - val_recall: 0.7330 - val_f1_score: 0.7491
Epoch 40/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3948 -
categorical_accuracy: 0.8464 - precision: 0.8621 - recall: 0.8298 - f1_score:
0.8458 - val_loss: 0.6917 - val_categorical_accuracy: 0.7423 - val_precision:
```

```
0.7567 - val_recall: 0.7207 - val_f1_score: 0.7383
Epoch 41/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3794 -
categorical_accuracy: 0.8522 - precision: 0.8666 - recall: 0.8368 - f1_score:
0.8517 - val_loss: 0.6801 - val_categorical_accuracy: 0.7523 - val_precision:
0.7653 - val_recall: 0.7360 - val_f1_score: 0.7489
Epoch 42/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3830 -
categorical_accuracy: 0.8497 - precision: 0.8647 - recall: 0.8338 - f1_score:
0.8492 - val_loss: 0.6793 - val_categorical_accuracy: 0.7467 - val_precision:
0.7626 - val_recall: 0.7293 - val_f1_score: 0.7443
Epoch 43/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3889 -
categorical_accuracy: 0.8487 - precision: 0.8636 - recall: 0.8321 - f1_score:
0.8483 - val_loss: 0.7009 - val_categorical_accuracy: 0.7393 - val_precision:
0.7593 - val_recall: 0.7185 - val_f1_score: 0.7360
Epoch 44/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3807 -
categorical_accuracy: 0.8527 - precision: 0.8672 - recall: 0.8366 - f1_score:
0.8521 - val_loss: 0.6641 - val_categorical_accuracy: 0.7452 - val_precision:
0.7636 - val_recall: 0.7325 - val_f1_score: 0.7435
Epoch 45/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3784 -
categorical_accuracy: 0.8516 - precision: 0.8665 - recall: 0.8356 - f1_score:
0.8511 - val_loss: 0.6777 - val_categorical_accuracy: 0.7515 - val_precision:
0.7688 - val_recall: 0.7340 - val_f1_score: 0.7497
Epoch 46/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3770 -
categorical_accuracy: 0.8547 - precision: 0.8692 - recall: 0.8380 - f1_score:
0.8542 - val_loss: 0.6709 - val_categorical_accuracy: 0.7475 - val_precision:
0.7645 - val_recall: 0.7295 - val_f1_score: 0.7457
Epoch 47/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3776 -
categorical_accuracy: 0.8533 - precision: 0.8678 - recall: 0.8363 - f1_score:
0.8528 - val_loss: 0.6708 - val_categorical_accuracy: 0.7550 - val_precision:
0.7709 - val_recall: 0.7362 - val_f1_score: 0.7521
Epoch 48/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3764 -
categorical_accuracy: 0.8541 - precision: 0.8695 - recall: 0.8377 - f1_score:
0.8536 - val_loss: 0.6593 - val_categorical_accuracy: 0.7642 - val_precision:
0.7766 - val_recall: 0.7450 - val_f1_score: 0.7623
Epoch 49/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3793 -
categorical_accuracy: 0.8527 - precision: 0.8681 - recall: 0.8363 - f1_score:
0.8523 - val_loss: 0.6766 - val_categorical_accuracy: 0.7500 - val_precision:
0.7680 - val_recall: 0.7318 - val_f1_score: 0.7487
Epoch 50/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3687 -
```

```
categorical_accuracy: 0.8586 - precision: 0.8718 - recall: 0.8426 - f1_score:
0.8581 - val_loss: 0.7147 - val_categorical_accuracy: 0.7492 - val_precision:
0.7629 - val_recall: 0.7327 - val_f1_score: 0.7468
Epoch 51/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3783 -
categorical_accuracy: 0.8537 - precision: 0.8681 - recall: 0.8370 - f1_score:
0.8533 - val_loss: 0.6693 - val_categorical_accuracy: 0.7552 - val_precision:
0.7685 - val_recall: 0.7370 - val_f1_score: 0.7527
Epoch 52/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3842 -
categorical_accuracy: 0.8518 - precision: 0.8665 - recall: 0.8353 - f1_score:
0.8512 - val_loss: 0.6670 - val_categorical_accuracy: 0.7533 - val_precision:
0.7754 - val_recall: 0.7355 - val_f1_score: 0.7518
Epoch 53/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3777 -
categorical_accuracy: 0.8538 - precision: 0.8699 - recall: 0.8367 - f1_score:
0.8534 - val_loss: 0.6990 - val_categorical_accuracy: 0.7485 - val_precision:
0.7638 - val_recall: 0.7322 - val_f1_score: 0.7444
Epoch 54/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3854 -
categorical_accuracy: 0.8482 - precision: 0.8657 - recall: 0.8310 - f1_score:
0.8476 - val_loss: 0.6619 - val_categorical_accuracy: 0.7520 - val_precision:
0.7737 - val_recall: 0.7315 - val_f1_score: 0.7508
Epoch 55/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3746 -
categorical_accuracy: 0.8548 - precision: 0.8707 - recall: 0.8370 - f1_score:
0.8544 - val_loss: 0.6740 - val_categorical_accuracy: 0.7560 - val_precision:
0.7731 - val_recall: 0.7350 - val_f1_score: 0.7539
Epoch 56/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3816 -
categorical_accuracy: 0.8510 - precision: 0.8658 - recall: 0.8352 - f1_score:
0.8505 - val_loss: 0.6610 - val_categorical_accuracy: 0.7598 - val_precision:
0.7744 - val_recall: 0.7408 - val_f1_score: 0.7592
Epoch 57/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3825 -
categorical_accuracy: 0.8493 - precision: 0.8652 - recall: 0.8342 - f1_score:
0.8487 - val_loss: 0.6710 - val_categorical_accuracy: 0.7520 - val_precision:
0.7720 - val_recall: 0.7347 - val_f1_score: 0.7506
Epoch 58/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3831 -
categorical_accuracy: 0.8501 - precision: 0.8648 - recall: 0.8335 - f1_score:
0.8497 - val_loss: 0.6708 - val_categorical_accuracy: 0.7538 - val_precision:
0.7688 - val_recall: 0.7358 - val_f1_score: 0.7524
Epoch 59/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3843 -
categorical_accuracy: 0.8507 - precision: 0.8661 - recall: 0.8327 - f1_score:
0.8502 - val_loss: 0.6671 - val_categorical_accuracy: 0.7560 - val_precision:
0.7729 - val_recall: 0.7385 - val_f1_score: 0.7539
```

```
Epoch 60/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3848 -
categorical_accuracy: 0.8478 - precision: 0.8632 - recall: 0.8314 - f1_score:
0.8473 - val_loss: 0.6728 - val_categorical_accuracy: 0.7500 - val_precision:
0.7666 - val_recall: 0.7318 - val_f1_score: 0.7481
Epoch 61/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3806 -
categorical_accuracy: 0.8503 - precision: 0.8658 - recall: 0.8333 - f1_score:
0.8498 - val_loss: 0.6715 - val_categorical_accuracy: 0.7552 - val_precision:
0.7684 - val_recall: 0.7368 - val_f1_score: 0.7542
Epoch 62/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3691 -
categorical_accuracy: 0.8560 - precision: 0.8711 - recall: 0.8402 - f1_score:
0.8557 - val_loss: 0.6572 - val_categorical_accuracy: 0.7623 - val_precision:
0.7763 - val_recall: 0.7452 - val_f1_score: 0.7606
Epoch 63/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3789 -
categorical_accuracy: 0.8518 - precision: 0.8662 - recall: 0.8358 - f1_score:
0.8513 - val_loss: 0.6565 - val_categorical_accuracy: 0.7542 - val_precision:
0.7728 - val_recall: 0.7383 - val_f1_score: 0.7519
Epoch 64/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3731 -
categorical_accuracy: 0.8556 - precision: 0.8709 - recall: 0.8385 - f1_score:
0.8552 - val_loss: 0.6611 - val_categorical_accuracy: 0.7527 - val_precision:
0.7723 - val_recall: 0.7300 - val_f1_score: 0.7514
Epoch 65/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3724 -
categorical_accuracy: 0.8549 - precision: 0.8709 - recall: 0.8391 - f1_score:
0.8545 - val_loss: 0.6602 - val_categorical_accuracy: 0.7585 - val_precision:
0.7775 - val_recall: 0.7380 - val_f1_score: 0.7588
Epoch 66/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3689 -
categorical_accuracy: 0.8556 - precision: 0.8704 - recall: 0.8405 - f1_score:
0.8552 - val_loss: 0.6705 - val_categorical_accuracy: 0.7582 - val_precision:
0.7761 - val_recall: 0.7452 - val_f1_score: 0.7568
Epoch 67/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3688 -
categorical_accuracy: 0.8546 - precision: 0.8696 - recall: 0.8392 - f1_score:
0.8540 - val_loss: 0.6602 - val_categorical_accuracy: 0.7630 - val_precision:
0.7783 - val_recall: 0.7452 - val_f1_score: 0.7613
Epoch 68/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3657 -
categorical_accuracy: 0.8579 - precision: 0.8711 - recall: 0.8437 - f1_score:
0.8576 - val_loss: 0.6809 - val_categorical_accuracy: 0.7450 - val_precision:
0.7621 - val_recall: 0.7280 - val_f1_score: 0.7430
Epoch 69/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3696 -
categorical_accuracy: 0.8576 - precision: 0.8710 - recall: 0.8420 - f1_score:
```

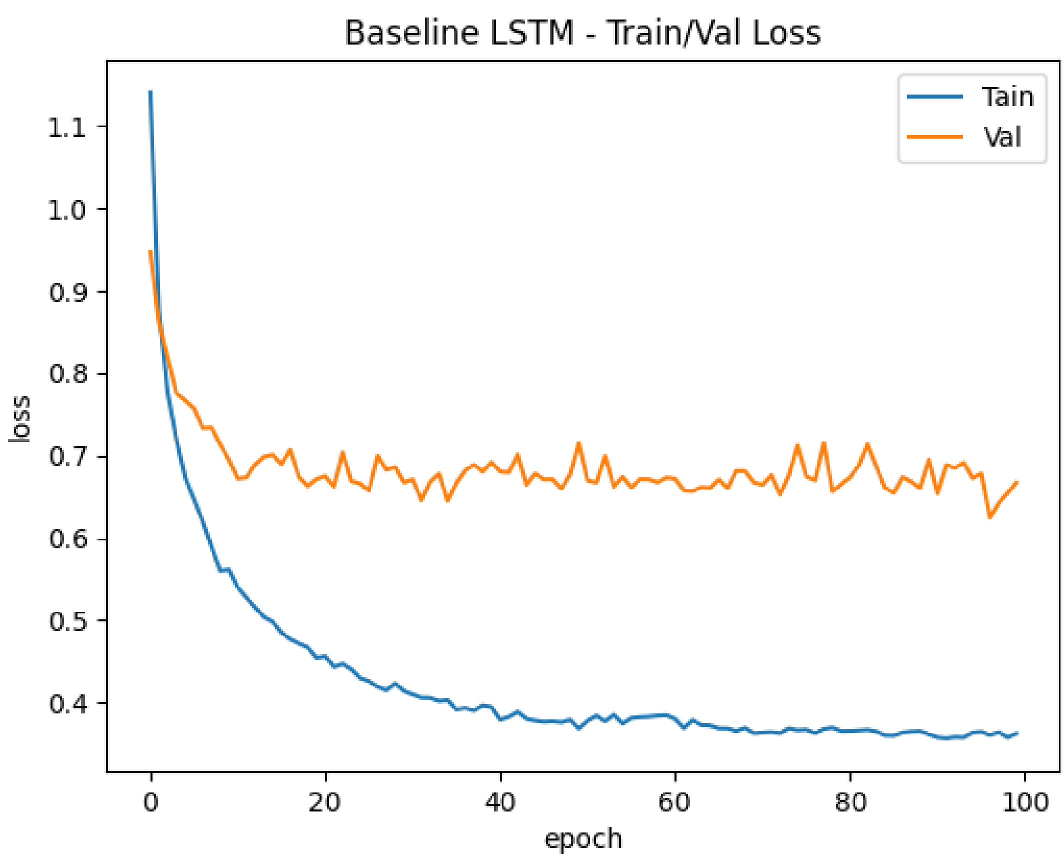
```
0.8572 - val_loss: 0.6807 - val_categorical_accuracy: 0.7542 - val_precision:
0.7715 - val_recall: 0.7395 - val_f1_score: 0.7518
Epoch 70/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3628 -
categorical_accuracy: 0.8546 - precision: 0.8706 - recall: 0.8402 - f1_score:
0.8542 - val_loss: 0.6669 - val_categorical_accuracy: 0.7600 - val_precision:
0.7773 - val_recall: 0.7435 - val_f1_score: 0.7575
Epoch 71/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3637 -
categorical_accuracy: 0.8564 - precision: 0.8704 - recall: 0.8423 - f1_score:
0.8560 - val_loss: 0.6642 - val_categorical_accuracy: 0.7523 - val_precision:
0.7683 - val_recall: 0.7355 - val_f1_score: 0.7502
Epoch 72/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3643 -
categorical_accuracy: 0.8581 - precision: 0.8727 - recall: 0.8422 - f1_score:
0.8577 - val_loss: 0.6759 - val_categorical_accuracy: 0.7498 - val_precision:
0.7643 - val_recall: 0.7295 - val_f1_score: 0.7477
Epoch 73/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3629 -
categorical_accuracy: 0.8595 - precision: 0.8740 - recall: 0.8433 - f1_score:
0.8591 - val_loss: 0.6520 - val_categorical_accuracy: 0.7567 - val_precision:
0.7729 - val_recall: 0.7350 - val_f1_score: 0.7549
Epoch 74/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3690 -
categorical_accuracy: 0.8533 - precision: 0.8673 - recall: 0.8381 - f1_score:
0.8527 - val_loss: 0.6755 - val_categorical_accuracy: 0.7550 - val_precision:
0.7716 - val_recall: 0.7375 - val_f1_score: 0.7510
Epoch 75/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3668 -
categorical_accuracy: 0.8590 - precision: 0.8736 - recall: 0.8432 - f1_score:
0.8586 - val_loss: 0.7118 - val_categorical_accuracy: 0.7435 - val_precision:
0.7572 - val_recall: 0.7253 - val_f1_score: 0.7403
Epoch 76/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3675 -
categorical_accuracy: 0.8570 - precision: 0.8731 - recall: 0.8412 - f1_score:
0.8566 - val_loss: 0.6743 - val_categorical_accuracy: 0.7515 - val_precision:
0.7664 - val_recall: 0.7347 - val_f1_score: 0.7500
Epoch 77/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3629 -
categorical_accuracy: 0.8587 - precision: 0.8715 - recall: 0.8437 - f1_score:
0.8583 - val_loss: 0.6696 - val_categorical_accuracy: 0.7502 - val_precision:
0.7683 - val_recall: 0.7370 - val_f1_score: 0.7485
Epoch 78/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3682 -
categorical_accuracy: 0.8556 - precision: 0.8697 - recall: 0.8408 - f1_score:
0.8551 - val_loss: 0.7149 - val_categorical_accuracy: 0.7533 - val_precision:
0.7700 - val_recall: 0.7358 - val_f1_score: 0.7496
Epoch 79/100
```

```
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3698 -
categorical_accuracy: 0.8578 - precision: 0.8723 - recall: 0.8422 - f1_score:
0.8573 - val_loss: 0.6560 - val_categorical_accuracy: 0.7635 - val_precision:
0.7789 - val_recall: 0.7477 - val_f1_score: 0.7605
Epoch 80/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3655 -
categorical_accuracy: 0.8574 - precision: 0.8718 - recall: 0.8424 - f1_score:
0.8571 - val_loss: 0.6651 - val_categorical_accuracy: 0.7598 - val_precision:
0.7784 - val_recall: 0.7420 - val_f1_score: 0.7581
Epoch 81/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3657 -
categorical_accuracy: 0.8597 - precision: 0.8735 - recall: 0.8446 - f1_score:
0.8593 - val_loss: 0.6732 - val_categorical_accuracy: 0.7452 - val_precision:
0.7639 - val_recall: 0.7320 - val_f1_score: 0.7442
Epoch 82/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3664 -
categorical_accuracy: 0.8593 - precision: 0.8733 - recall: 0.8438 - f1_score:
0.8589 - val_loss: 0.6875 - val_categorical_accuracy: 0.7517 - val_precision:
0.7687 - val_recall: 0.7362 - val_f1_score: 0.7491
Epoch 83/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3672 -
categorical_accuracy: 0.8576 - precision: 0.8710 - recall: 0.8428 - f1_score:
0.8572 - val_loss: 0.7132 - val_categorical_accuracy: 0.7500 - val_precision:
0.7643 - val_recall: 0.7320 - val_f1_score: 0.7465
Epoch 84/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3651 -
categorical_accuracy: 0.8603 - precision: 0.8748 - recall: 0.8462 - f1_score:
0.8599 - val_loss: 0.6857 - val_categorical_accuracy: 0.7542 - val_precision:
0.7702 - val_recall: 0.7347 - val_f1_score: 0.7524
Epoch 85/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3604 -
categorical_accuracy: 0.8597 - precision: 0.8733 - recall: 0.8459 - f1_score:
0.8593 - val_loss: 0.6604 - val_categorical_accuracy: 0.7617 - val_precision:
0.7796 - val_recall: 0.7445 - val_f1_score: 0.7595
Epoch 86/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3601 -
categorical_accuracy: 0.8595 - precision: 0.8736 - recall: 0.8458 - f1_score:
0.8592 - val_loss: 0.6543 - val_categorical_accuracy: 0.7598 - val_precision:
0.7767 - val_recall: 0.7427 - val_f1_score: 0.7578
Epoch 87/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3638 -
categorical_accuracy: 0.8575 - precision: 0.8720 - recall: 0.8425 - f1_score:
0.8571 - val_loss: 0.6736 - val_categorical_accuracy: 0.7580 - val_precision:
0.7787 - val_recall: 0.7408 - val_f1_score: 0.7559
Epoch 88/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3650 -
categorical_accuracy: 0.8581 - precision: 0.8720 - recall: 0.8426 - f1_score:
0.8577 - val_loss: 0.6679 - val_categorical_accuracy: 0.7530 - val_precision:
```

```
0.7715 - val_recall: 0.7370 - val_f1_score: 0.7520
Epoch 89/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3655 -
categorical_accuracy: 0.8579 - precision: 0.8725 - recall: 0.8428 - f1_score:
0.8575 - val_loss: 0.6603 - val_categorical_accuracy: 0.7623 - val_precision:
0.7799 - val_recall: 0.7458 - val_f1_score: 0.7603
Epoch 90/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3616 -
categorical_accuracy: 0.8607 - precision: 0.8759 - recall: 0.8451 - f1_score:
0.8603 - val_loss: 0.6948 - val_categorical_accuracy: 0.7487 - val_precision:
0.7676 - val_recall: 0.7347 - val_f1_score: 0.7463
Epoch 91/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3579 -
categorical_accuracy: 0.8616 - precision: 0.8737 - recall: 0.8473 - f1_score:
0.8611 - val_loss: 0.6535 - val_categorical_accuracy: 0.7615 - val_precision:
0.7770 - val_recall: 0.7405 - val_f1_score: 0.7594
Epoch 92/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3568 -
categorical_accuracy: 0.8622 - precision: 0.8760 - recall: 0.8481 - f1_score:
0.8618 - val_loss: 0.6882 - val_categorical_accuracy: 0.7577 - val_precision:
0.7709 - val_recall: 0.7412 - val_f1_score: 0.7557
Epoch 93/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3583 -
categorical_accuracy: 0.8614 - precision: 0.8747 - recall: 0.8460 - f1_score:
0.8610 - val_loss: 0.6846 - val_categorical_accuracy: 0.7545 - val_precision:
0.7680 - val_recall: 0.7398 - val_f1_score: 0.7531
Epoch 94/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3579 -
categorical_accuracy: 0.8602 - precision: 0.8733 - recall: 0.8464 - f1_score:
0.8598 - val_loss: 0.6911 - val_categorical_accuracy: 0.7655 - val_precision:
0.7809 - val_recall: 0.7477 - val_f1_score: 0.7599
Epoch 95/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3637 -
categorical_accuracy: 0.8581 - precision: 0.8730 - recall: 0.8424 - f1_score:
0.8577 - val_loss: 0.6722 - val_categorical_accuracy: 0.7598 - val_precision:
0.7765 - val_recall: 0.7435 - val_f1_score: 0.7563
Epoch 96/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3647 -
categorical_accuracy: 0.8580 - precision: 0.8728 - recall: 0.8435 - f1_score:
0.8576 - val_loss: 0.6777 - val_categorical_accuracy: 0.7588 - val_precision:
0.7776 - val_recall: 0.7405 - val_f1_score: 0.7545
Epoch 97/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3607 -
categorical_accuracy: 0.8601 - precision: 0.8746 - recall: 0.8460 - f1_score:
0.8597 - val_loss: 0.6243 - val_categorical_accuracy: 0.7740 - val_precision:
0.7912 - val_recall: 0.7598 - val_f1_score: 0.7718
Epoch 98/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3642 -
```

```
categorical_accuracy: 0.8609 - precision: 0.8749 - recall: 0.8454 - f1_score:
0.8605 - val_loss: 0.6422 - val_categorical_accuracy: 0.7648 - val_precision:
0.7795 - val_recall: 0.7450 - val_f1_score: 0.7621
Epoch 99/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3579 -
categorical_accuracy: 0.8601 - precision: 0.8746 - recall: 0.8461 - f1_score:
0.8597 - val_loss: 0.6541 - val_categorical_accuracy: 0.7645 - val_precision:
0.7835 - val_recall: 0.7427 - val_f1_score: 0.7636
Epoch 100/100
1000/1000 [==============================] - 12s 12ms/step - loss: 0.3625 -
categorical_accuracy: 0.8594 - precision: 0.8738 - recall: 0.8443 - f1_score:
0.8589 - val_loss: 0.6667 - val_categorical_accuracy: 0.7648 - val_precision:
0.7835 - val_recall: 0.7508 - val_f1_score: 0.7633
```

[ ]:
```
#plot loss
plt.plot(history_lstm.history['loss'])
plt.plot(history_lstm.history['val_loss'])
plt.title('Baseline LSTM - Train/Val Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Tain', 'Val'])
plt.show()
```



Observations on Baseline LSTM As illustrated in the loss curves above, the Baseline LSTM is overfitting on the training data. Our next LSTM will need to incorporate techniques to mitigate this.

```
[ ]: # free up resources
gc.collect()
```

```
[ ]: 0
```

Improved LSTM Model 1 This LSTM will add improvements to the baseline model including additional hidden LSTM layers (2) and start with a lower number of units per layer (128). The second LSTM layer will reduce the number of units to 32. An additional fully connected layer is added. Dropout is also added to both the convolutional layers and the fully connected layers.

```
[ ]: checkpoint_filepath = '/content/drive/MyDrive/USD/models/composer-classifier/
  ↪lstm-1'
model_checkpoint_callback = keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    monitor='val_categorical_accuracy',
    mode='max',
    save_best_only=True)

# define LSTM with mitigations for overfitting: more layers, more dropout
lstm_exp1 = tf.keras.models.Sequential([

    # input our sequence tensors
    tf.keras.layers.Input(shape=(NORM_SEQUENCE_LENGTH, NUM_PIANO_KEYS)),
    tf.keras.layers.Normalization(axis=None),
    tf.keras.layers.LSTM(128, return_sequences=True),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.LSTM(32, return_sequences=False),

    # classification head
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(units = NUM_COMPOSERS, activation='softmax')
])

# Compile the model
lstm_exp1.compile(
    optimizer=keras.optimizers.Adam(learning_rate=LEARNING_RATE),
    loss=tf.keras.losses.CategoricalCrossentropy(),
```

```
    metrics=[keras.metrics.CategoricalAccuracy(), keras.metrics.Precision(),␣
 ↪keras.metrics.Recall(), keras.metrics.F1Score()]
)
```

[ ]:
```
# Train the model
history_lstm_exp1 = lstm_exp1.fit(X_train, y_train,␣
 ↪validation_data=(X_val,y_val), epochs=NUM_EPOCHS, batch_size=BATCH_SIZE,␣
 ↪callbacks=[model_checkpoint_callback])
```

```
Epoch 1/75
1200/1200 [==============================] - 31s 22ms/step - loss: 1.2037 -
categorical_accuracy: 0.4629 - precision_2: 0.6200 - recall_2: 0.1871 -
f1_score: 0.4605 - val_loss: 1.0805 - val_categorical_accuracy: 0.5410 -
val_precision_2: 0.6563 - val_recall_2: 0.3179 - val_f1_score: 0.5341
Epoch 2/75
1200/1200 [==============================] - 26s 21ms/step - loss: 1.0342 -
categorical_accuracy: 0.5687 - precision_2: 0.6524 - recall_2: 0.3936 -
f1_score: 0.5566 - val_loss: 0.9727 - val_categorical_accuracy: 0.5919 -
val_precision_2: 0.6636 - val_recall_2: 0.4660 - val_f1_score: 0.5726
Epoch 3/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.9164 -
categorical_accuracy: 0.6264 - precision_2: 0.6946 - recall_2: 0.5091 -
f1_score: 0.6183 - val_loss: 0.9109 - val_categorical_accuracy: 0.6233 -
val_precision_2: 0.6979 - val_recall_2: 0.5025 - val_f1_score: 0.6105
Epoch 4/75
1200/1200 [==============================] - 26s 22ms/step - loss: 0.8160 -
categorical_accuracy: 0.6692 - precision_2: 0.7254 - recall_2: 0.5810 -
f1_score: 0.6629 - val_loss: 0.7820 - val_categorical_accuracy: 0.6762 -
val_precision_2: 0.7355 - val_recall_2: 0.5938 - val_f1_score: 0.6760
Epoch 5/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.7168 -
categorical_accuracy: 0.7131 - precision_2: 0.7580 - recall_2: 0.6500 -
f1_score: 0.7080 - val_loss: 0.6930 - val_categorical_accuracy: 0.7258 -
val_precision_2: 0.7635 - val_recall_2: 0.6740 - val_f1_score: 0.7241
Epoch 6/75
1200/1200 [==============================] - 26s 21ms/step - loss: 0.6218 -
categorical_accuracy: 0.7565 - precision_2: 0.7860 - recall_2: 0.7157 -
f1_score: 0.7534 - val_loss: 0.6560 - val_categorical_accuracy: 0.7377 -
val_precision_2: 0.7742 - val_recall_2: 0.6888 - val_f1_score: 0.7398
Epoch 7/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.5534 -
categorical_accuracy: 0.7856 - precision_2: 0.8102 - recall_2: 0.7551 -
f1_score: 0.7836 - val_loss: 0.6100 - val_categorical_accuracy: 0.7594 -
val_precision_2: 0.7911 - val_recall_2: 0.7273 - val_f1_score: 0.7604
Epoch 8/75
1200/1200 [==============================] - 26s 22ms/step - loss: 0.5032 -
categorical_accuracy: 0.8093 - precision_2: 0.8302 - recall_2: 0.7836 -
f1_score: 0.8076 - val_loss: 0.5962 - val_categorical_accuracy: 0.7640 -
```

```
val_precision_2: 0.7895 - val_recall_2: 0.7385 - val_f1_score: 0.7637
Epoch 9/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.4572 -
categorical_accuracy: 0.8278 - precision_2: 0.8466 - recall_2: 0.8073 -
f1_score: 0.8267 - val_loss: 0.5420 - val_categorical_accuracy: 0.7896 -
val_precision_2: 0.8151 - val_recall_2: 0.7671 - val_f1_score: 0.7866
Epoch 10/75
1200/1200 [==============================] - 26s 21ms/step - loss: 0.4232 -
categorical_accuracy: 0.8397 - precision_2: 0.8567 - recall_2: 0.8219 -
f1_score: 0.8388 - val_loss: 0.5263 - val_categorical_accuracy: 0.7921 -
val_precision_2: 0.8129 - val_recall_2: 0.7723 - val_f1_score: 0.7915
Epoch 11/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.3936 -
categorical_accuracy: 0.8552 - precision_2: 0.8700 - recall_2: 0.8398 -
f1_score: 0.8545 - val_loss: 0.5077 - val_categorical_accuracy: 0.8106 -
val_precision_2: 0.8358 - val_recall_2: 0.7754 - val_f1_score: 0.8119
Epoch 12/75
1200/1200 [==============================] - 20s 16ms/step - loss: 0.3759 -
categorical_accuracy: 0.8630 - precision_2: 0.8767 - recall_2: 0.8487 -
f1_score: 0.8624 - val_loss: 0.5302 - val_categorical_accuracy: 0.7979 -
val_precision_2: 0.8134 - val_recall_2: 0.7790 - val_f1_score: 0.8004
Epoch 13/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.3552 -
categorical_accuracy: 0.8730 - precision_2: 0.8841 - recall_2: 0.8598 -
f1_score: 0.8724 - val_loss: 0.4734 - val_categorical_accuracy: 0.8202 -
val_precision_2: 0.8409 - val_recall_2: 0.7971 - val_f1_score: 0.8192
Epoch 14/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.3365 -
categorical_accuracy: 0.8784 - precision_2: 0.8900 - recall_2: 0.8671 -
f1_score: 0.8779 - val_loss: 0.4584 - val_categorical_accuracy: 0.8290 -
val_precision_2: 0.8482 - val_recall_2: 0.8102 - val_f1_score: 0.8305
Epoch 15/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.3175 -
categorical_accuracy: 0.8843 - precision_2: 0.8949 - recall_2: 0.8742 -
f1_score: 0.8838 - val_loss: 0.4955 - val_categorical_accuracy: 0.8152 -
val_precision_2: 0.8261 - val_recall_2: 0.7998 - val_f1_score: 0.8176
Epoch 16/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.2963 -
categorical_accuracy: 0.8925 - precision_2: 0.9025 - recall_2: 0.8826 -
f1_score: 0.8922 - val_loss: 0.4778 - val_categorical_accuracy: 0.8198 -
val_precision_2: 0.8423 - val_recall_2: 0.8033 - val_f1_score: 0.8221
Epoch 17/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.2865 -
categorical_accuracy: 0.8971 - precision_2: 0.9073 - recall_2: 0.8884 -
f1_score: 0.8967 - val_loss: 0.4345 - val_categorical_accuracy: 0.8421 -
val_precision_2: 0.8586 - val_recall_2: 0.8233 - val_f1_score: 0.8404
Epoch 18/75
1200/1200 [==============================] - 20s 16ms/step - loss: 0.2767 -
```

```
categorical_accuracy: 0.9032 - precision_2: 0.9123 - recall_2: 0.8943 -
f1_score: 0.9029 - val_loss: 0.4620 - val_categorical_accuracy: 0.8329 -
val_precision_2: 0.8482 - val_recall_2: 0.8181 - val_f1_score: 0.8345
Epoch 19/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.2684 -
categorical_accuracy: 0.9051 - precision_2: 0.9129 - recall_2: 0.8967 -
f1_score: 0.9049 - val_loss: 0.4503 - val_categorical_accuracy: 0.8392 -
val_precision_2: 0.8560 - val_recall_2: 0.8250 - val_f1_score: 0.8399
Epoch 20/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.2528 -
categorical_accuracy: 0.9101 - precision_2: 0.9176 - recall_2: 0.9016 -
f1_score: 0.9098 - val_loss: 0.4266 - val_categorical_accuracy: 0.8410 -
val_precision_2: 0.8565 - val_recall_2: 0.8292 - val_f1_score: 0.8414
Epoch 21/75
1200/1200 [==============================] - 26s 21ms/step - loss: 0.2448 -
categorical_accuracy: 0.9135 - precision_2: 0.9216 - recall_2: 0.9067 -
f1_score: 0.9133 - val_loss: 0.4181 - val_categorical_accuracy: 0.8435 -
val_precision_2: 0.8581 - val_recall_2: 0.8317 - val_f1_score: 0.8430
Epoch 22/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.2420 -
categorical_accuracy: 0.9132 - precision_2: 0.9220 - recall_2: 0.9064 -
f1_score: 0.9130 - val_loss: 0.4704 - val_categorical_accuracy: 0.8210 -
val_precision_2: 0.8322 - val_recall_2: 0.8092 - val_f1_score: 0.8200
Epoch 23/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.2309 -
categorical_accuracy: 0.9173 - precision_2: 0.9246 - recall_2: 0.9108 -
f1_score: 0.9172 - val_loss: 0.4358 - val_categorical_accuracy: 0.8344 -
val_precision_2: 0.8499 - val_recall_2: 0.8177 - val_f1_score: 0.8357
Epoch 24/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.2255 -
categorical_accuracy: 0.9200 - precision_2: 0.9273 - recall_2: 0.9130 -
f1_score: 0.9198 - val_loss: 0.4799 - val_categorical_accuracy: 0.8206 -
val_precision_2: 0.8362 - val_recall_2: 0.8040 - val_f1_score: 0.8202
Epoch 25/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.2152 -
categorical_accuracy: 0.9240 - precision_2: 0.9308 - recall_2: 0.9186 -
f1_score: 0.9238 - val_loss: 0.3945 - val_categorical_accuracy: 0.8610 -
val_precision_2: 0.8732 - val_recall_2: 0.8481 - val_f1_score: 0.8622
Epoch 26/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.2083 -
categorical_accuracy: 0.9268 - precision_2: 0.9327 - recall_2: 0.9209 -
f1_score: 0.9266 - val_loss: 0.4022 - val_categorical_accuracy: 0.8567 -
val_precision_2: 0.8745 - val_recall_2: 0.8419 - val_f1_score: 0.8570
Epoch 27/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1988 -
categorical_accuracy: 0.9289 - precision_2: 0.9349 - recall_2: 0.9235 -
f1_score: 0.9288 - val_loss: 0.4156 - val_categorical_accuracy: 0.8552 -
val_precision_2: 0.8657 - val_recall_2: 0.8450 - val_f1_score: 0.8559
```

```
Epoch 28/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1988 -
categorical_accuracy: 0.9284 - precision_2: 0.9341 - recall_2: 0.9228 -
f1_score: 0.9283 - val_loss: 0.3999 - val_categorical_accuracy: 0.8606 -
val_precision_2: 0.8772 - val_recall_2: 0.8435 - val_f1_score: 0.8612
Epoch 29/75
1200/1200 [==============================] - 26s 22ms/step - loss: 0.1850 -
categorical_accuracy: 0.9360 - precision_2: 0.9407 - recall_2: 0.9308 -
f1_score: 0.9359 - val_loss: 0.3994 - val_categorical_accuracy: 0.8612 -
val_precision_2: 0.8718 - val_recall_2: 0.8475 - val_f1_score: 0.8617
Epoch 30/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.1916 -
categorical_accuracy: 0.9331 - precision_2: 0.9392 - recall_2: 0.9273 -
f1_score: 0.9330 - val_loss: 0.3884 - val_categorical_accuracy: 0.8658 -
val_precision_2: 0.8744 - val_recall_2: 0.8529 - val_f1_score: 0.8660
Epoch 31/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1812 -
categorical_accuracy: 0.9354 - precision_2: 0.9405 - recall_2: 0.9309 -
f1_score: 0.9353 - val_loss: 0.3858 - val_categorical_accuracy: 0.8648 -
val_precision_2: 0.8793 - val_recall_2: 0.8531 - val_f1_score: 0.8648
Epoch 32/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.1753 -
categorical_accuracy: 0.9387 - precision_2: 0.9439 - recall_2: 0.9334 -
f1_score: 0.9386 - val_loss: 0.3787 - val_categorical_accuracy: 0.8692 -
val_precision_2: 0.8785 - val_recall_2: 0.8583 - val_f1_score: 0.8701
Epoch 33/75
1200/1200 [==============================] - 20s 16ms/step - loss: 0.1724 -
categorical_accuracy: 0.9393 - precision_2: 0.9434 - recall_2: 0.9351 -
f1_score: 0.9393 - val_loss: 0.4026 - val_categorical_accuracy: 0.8596 -
val_precision_2: 0.8674 - val_recall_2: 0.8529 - val_f1_score: 0.8601
Epoch 34/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1695 -
categorical_accuracy: 0.9407 - precision_2: 0.9454 - recall_2: 0.9365 -
f1_score: 0.9407 - val_loss: 0.3935 - val_categorical_accuracy: 0.8633 -
val_precision_2: 0.8701 - val_recall_2: 0.8567 - val_f1_score: 0.8638
Epoch 35/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1679 -
categorical_accuracy: 0.9415 - precision_2: 0.9462 - recall_2: 0.9374 -
f1_score: 0.9414 - val_loss: 0.4012 - val_categorical_accuracy: 0.8627 -
val_precision_2: 0.8692 - val_recall_2: 0.8554 - val_f1_score: 0.8632
Epoch 36/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1597 -
categorical_accuracy: 0.9437 - precision_2: 0.9484 - recall_2: 0.9398 -
f1_score: 0.9436 - val_loss: 0.3764 - val_categorical_accuracy: 0.8654 -
val_precision_2: 0.8737 - val_recall_2: 0.8575 - val_f1_score: 0.8660
Epoch 37/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1583 -
categorical_accuracy: 0.9448 - precision_2: 0.9489 - recall_2: 0.9411 -
```

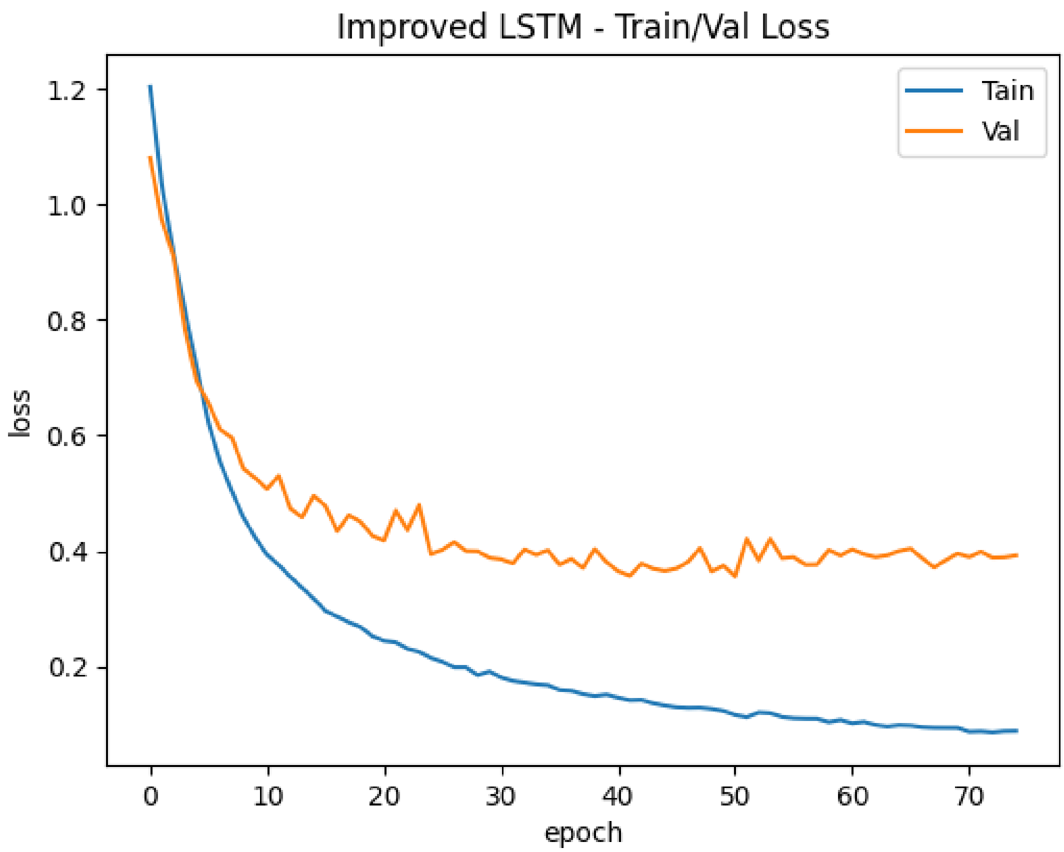
```
f1_score: 0.9448 - val_loss: 0.3863 - val_categorical_accuracy: 0.8610 -
val_precision_2: 0.8689 - val_recall_2: 0.8533 - val_f1_score: 0.8630
Epoch 38/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.1526 -
categorical_accuracy: 0.9475 - precision_2: 0.9511 - recall_2: 0.9441 -
f1_score: 0.9475 - val_loss: 0.3709 - val_categorical_accuracy: 0.8752 -
val_precision_2: 0.8837 - val_recall_2: 0.8677 - val_f1_score: 0.8753
Epoch 39/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1491 -
categorical_accuracy: 0.9489 - precision_2: 0.9524 - recall_2: 0.9454 -
f1_score: 0.9489 - val_loss: 0.4033 - val_categorical_accuracy: 0.8679 -
val_precision_2: 0.8766 - val_recall_2: 0.8596 - val_f1_score: 0.8691
Epoch 40/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1516 -
categorical_accuracy: 0.9470 - precision_2: 0.9507 - recall_2: 0.9434 -
f1_score: 0.9470 - val_loss: 0.3809 - val_categorical_accuracy: 0.8706 -
val_precision_2: 0.8787 - val_recall_2: 0.8606 - val_f1_score: 0.8710
Epoch 41/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1463 -
categorical_accuracy: 0.9489 - precision_2: 0.9525 - recall_2: 0.9455 -
f1_score: 0.9489 - val_loss: 0.3651 - val_categorical_accuracy: 0.8742 -
val_precision_2: 0.8818 - val_recall_2: 0.8671 - val_f1_score: 0.8737
Epoch 42/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1418 -
categorical_accuracy: 0.9508 - precision_2: 0.9546 - recall_2: 0.9474 -
f1_score: 0.9507 - val_loss: 0.3573 - val_categorical_accuracy: 0.8752 -
val_precision_2: 0.8846 - val_recall_2: 0.8675 - val_f1_score: 0.8758
Epoch 43/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1423 -
categorical_accuracy: 0.9509 - precision_2: 0.9537 - recall_2: 0.9473 -
f1_score: 0.9508 - val_loss: 0.3782 - val_categorical_accuracy: 0.8723 -
val_precision_2: 0.8812 - val_recall_2: 0.8619 - val_f1_score: 0.8731
Epoch 44/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.1366 -
categorical_accuracy: 0.9526 - precision_2: 0.9562 - recall_2: 0.9492 -
f1_score: 0.9526 - val_loss: 0.3695 - val_categorical_accuracy: 0.8779 -
val_precision_2: 0.8842 - val_recall_2: 0.8719 - val_f1_score: 0.8775
Epoch 45/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1327 -
categorical_accuracy: 0.9539 - precision_2: 0.9569 - recall_2: 0.9511 -
f1_score: 0.9538 - val_loss: 0.3656 - val_categorical_accuracy: 0.8758 -
val_precision_2: 0.8854 - val_recall_2: 0.8677 - val_f1_score: 0.8769
Epoch 46/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1297 -
categorical_accuracy: 0.9545 - precision_2: 0.9574 - recall_2: 0.9520 -
f1_score: 0.9545 - val_loss: 0.3699 - val_categorical_accuracy: 0.8771 -
val_precision_2: 0.8832 - val_recall_2: 0.8731 - val_f1_score: 0.8774
Epoch 47/75
```

```
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1287 -
categorical_accuracy: 0.9554 - precision_2: 0.9585 - recall_2: 0.9527 -
f1_score: 0.9554 - val_loss: 0.3815 - val_categorical_accuracy: 0.8660 -
val_precision_2: 0.8756 - val_recall_2: 0.8594 - val_f1_score: 0.8667
Epoch 48/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1291 -
categorical_accuracy: 0.9549 - precision_2: 0.9581 - recall_2: 0.9522 -
f1_score: 0.9549 - val_loss: 0.4050 - val_categorical_accuracy: 0.8656 -
val_precision_2: 0.8736 - val_recall_2: 0.8583 - val_f1_score: 0.8658
Epoch 49/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.1269 -
categorical_accuracy: 0.9554 - precision_2: 0.9590 - recall_2: 0.9526 -
f1_score: 0.9554 - val_loss: 0.3647 - val_categorical_accuracy: 0.8810 -
val_precision_2: 0.8912 - val_recall_2: 0.8717 - val_f1_score: 0.8812
Epoch 50/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1234 -
categorical_accuracy: 0.9567 - precision_2: 0.9598 - recall_2: 0.9538 -
f1_score: 0.9566 - val_loss: 0.3748 - val_categorical_accuracy: 0.8777 -
val_precision_2: 0.8849 - val_recall_2: 0.8694 - val_f1_score: 0.8781
Epoch 51/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1164 -
categorical_accuracy: 0.9603 - precision_2: 0.9626 - recall_2: 0.9584 -
f1_score: 0.9602 - val_loss: 0.3560 - val_categorical_accuracy: 0.8800 -
val_precision_2: 0.8855 - val_recall_2: 0.8746 - val_f1_score: 0.8797
Epoch 52/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1124 -
categorical_accuracy: 0.9610 - precision_2: 0.9633 - recall_2: 0.9590 -
f1_score: 0.9610 - val_loss: 0.4213 - val_categorical_accuracy: 0.8600 -
val_precision_2: 0.8692 - val_recall_2: 0.8540 - val_f1_score: 0.8616
Epoch 53/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1208 -
categorical_accuracy: 0.9575 - precision_2: 0.9604 - recall_2: 0.9551 -
f1_score: 0.9575 - val_loss: 0.3833 - val_categorical_accuracy: 0.8758 -
val_precision_2: 0.8828 - val_recall_2: 0.8706 - val_f1_score: 0.8759
Epoch 54/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1196 -
categorical_accuracy: 0.9579 - precision_2: 0.9606 - recall_2: 0.9555 -
f1_score: 0.9579 - val_loss: 0.4214 - val_categorical_accuracy: 0.8602 -
val_precision_2: 0.8687 - val_recall_2: 0.8548 - val_f1_score: 0.8615
Epoch 55/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1130 -
categorical_accuracy: 0.9617 - precision_2: 0.9643 - recall_2: 0.9592 -
f1_score: 0.9617 - val_loss: 0.3876 - val_categorical_accuracy: 0.8665 -
val_precision_2: 0.8761 - val_recall_2: 0.8590 - val_f1_score: 0.8674
Epoch 56/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1107 -
categorical_accuracy: 0.9613 - precision_2: 0.9634 - recall_2: 0.9594 -
f1_score: 0.9613 - val_loss: 0.3896 - val_categorical_accuracy: 0.8758 -
```

```
val_precision_2: 0.8826 - val_recall_2: 0.8696 - val_f1_score: 0.8762
Epoch 57/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1100 -
categorical_accuracy: 0.9620 - precision_2: 0.9643 - recall_2: 0.9596 -
f1_score: 0.9620 - val_loss: 0.3763 - val_categorical_accuracy: 0.8796 -
val_precision_2: 0.8844 - val_recall_2: 0.8737 - val_f1_score: 0.8804
Epoch 58/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1099 -
categorical_accuracy: 0.9628 - precision_2: 0.9653 - recall_2: 0.9604 -
f1_score: 0.9628 - val_loss: 0.3766 - val_categorical_accuracy: 0.8702 -
val_precision_2: 0.8766 - val_recall_2: 0.8640 - val_f1_score: 0.8712
Epoch 59/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1040 -
categorical_accuracy: 0.9641 - precision_2: 0.9662 - recall_2: 0.9624 -
f1_score: 0.9641 - val_loss: 0.4018 - val_categorical_accuracy: 0.8642 -
val_precision_2: 0.8702 - val_recall_2: 0.8575 - val_f1_score: 0.8654
Epoch 60/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1077 -
categorical_accuracy: 0.9621 - precision_2: 0.9642 - recall_2: 0.9601 -
f1_score: 0.9621 - val_loss: 0.3922 - val_categorical_accuracy: 0.8773 -
val_precision_2: 0.8833 - val_recall_2: 0.8706 - val_f1_score: 0.8775
Epoch 61/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1020 -
categorical_accuracy: 0.9645 - precision_2: 0.9667 - recall_2: 0.9627 -
f1_score: 0.9645 - val_loss: 0.4029 - val_categorical_accuracy: 0.8700 -
val_precision_2: 0.8761 - val_recall_2: 0.8658 - val_f1_score: 0.8709
Epoch 62/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.1044 -
categorical_accuracy: 0.9645 - precision_2: 0.9663 - recall_2: 0.9623 -
f1_score: 0.9644 - val_loss: 0.3945 - val_categorical_accuracy: 0.8721 -
val_precision_2: 0.8781 - val_recall_2: 0.8673 - val_f1_score: 0.8727
Epoch 63/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0991 -
categorical_accuracy: 0.9651 - precision_2: 0.9672 - recall_2: 0.9635 -
f1_score: 0.9651 - val_loss: 0.3895 - val_categorical_accuracy: 0.8742 -
val_precision_2: 0.8791 - val_recall_2: 0.8694 - val_f1_score: 0.8750
Epoch 64/75
1200/1200 [==============================] - 20s 16ms/step - loss: 0.0963 -
categorical_accuracy: 0.9663 - precision_2: 0.9683 - recall_2: 0.9645 -
f1_score: 0.9663 - val_loss: 0.3927 - val_categorical_accuracy: 0.8735 -
val_precision_2: 0.8797 - val_recall_2: 0.8685 - val_f1_score: 0.8744
Epoch 65/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0988 -
categorical_accuracy: 0.9658 - precision_2: 0.9678 - recall_2: 0.9637 -
f1_score: 0.9658 - val_loss: 0.3999 - val_categorical_accuracy: 0.8692 -
val_precision_2: 0.8773 - val_recall_2: 0.8642 - val_f1_score: 0.8692
Epoch 66/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0980 -
```

```
categorical_accuracy: 0.9661 - precision_2: 0.9682 - recall_2: 0.9645 -
f1_score: 0.9661 - val_loss: 0.4042 - val_categorical_accuracy: 0.8669 -
val_precision_2: 0.8730 - val_recall_2: 0.8623 - val_f1_score: 0.8681
Epoch 67/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0953 -
categorical_accuracy: 0.9668 - precision_2: 0.9686 - recall_2: 0.9654 -
f1_score: 0.9668 - val_loss: 0.3872 - val_categorical_accuracy: 0.8744 -
val_precision_2: 0.8764 - val_recall_2: 0.8698 - val_f1_score: 0.8752
Epoch 68/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.0944 -
categorical_accuracy: 0.9682 - precision_2: 0.9699 - recall_2: 0.9665 -
f1_score: 0.9682 - val_loss: 0.3716 - val_categorical_accuracy: 0.8815 -
val_precision_2: 0.8880 - val_recall_2: 0.8767 - val_f1_score: 0.8820
Epoch 69/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0942 -
categorical_accuracy: 0.9673 - precision_2: 0.9693 - recall_2: 0.9658 -
f1_score: 0.9673 - val_loss: 0.3841 - val_categorical_accuracy: 0.8775 -
val_precision_2: 0.8843 - val_recall_2: 0.8742 - val_f1_score: 0.8773
Epoch 70/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0941 -
categorical_accuracy: 0.9675 - precision_2: 0.9693 - recall_2: 0.9660 -
f1_score: 0.9675 - val_loss: 0.3956 - val_categorical_accuracy: 0.8788 -
val_precision_2: 0.8843 - val_recall_2: 0.8758 - val_f1_score: 0.8793
Epoch 71/75
1200/1200 [==============================] - 25s 21ms/step - loss: 0.0875 -
categorical_accuracy: 0.9695 - precision_2: 0.9713 - recall_2: 0.9678 -
f1_score: 0.9694 - val_loss: 0.3898 - val_categorical_accuracy: 0.8867 -
val_precision_2: 0.8912 - val_recall_2: 0.8823 - val_f1_score: 0.8874
Epoch 72/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0881 -
categorical_accuracy: 0.9699 - precision_2: 0.9715 - recall_2: 0.9685 -
f1_score: 0.9699 - val_loss: 0.3988 - val_categorical_accuracy: 0.8792 -
val_precision_2: 0.8843 - val_recall_2: 0.8756 - val_f1_score: 0.8792
Epoch 73/75
1200/1200 [==============================] - 26s 21ms/step - loss: 0.0864 -
categorical_accuracy: 0.9705 - precision_2: 0.9721 - recall_2: 0.9694 -
f1_score: 0.9705 - val_loss: 0.3886 - val_categorical_accuracy: 0.8871 -
val_precision_2: 0.8922 - val_recall_2: 0.8846 - val_f1_score: 0.8867
Epoch 74/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0884 -
categorical_accuracy: 0.9692 - precision_2: 0.9712 - recall_2: 0.9678 -
f1_score: 0.9692 - val_loss: 0.3891 - val_categorical_accuracy: 0.8806 -
val_precision_2: 0.8854 - val_recall_2: 0.8775 - val_f1_score: 0.8807
Epoch 75/75
1200/1200 [==============================] - 19s 16ms/step - loss: 0.0887 -
categorical_accuracy: 0.9703 - precision_2: 0.9717 - recall_2: 0.9687 -
f1_score: 0.9703 - val_loss: 0.3924 - val_categorical_accuracy: 0.8773 -
val_precision_2: 0.8815 - val_recall_2: 0.8740 - val_f1_score: 0.8776
```

[ ]:
```
#plot loss
plt.plot(history_lstm_exp1.history['loss'])
plt.plot(history_lstm_exp1.history['val_loss'])
plt.title('Improved LSTM - Train/Val Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Tain', 'Val'])
plt.show()
```



[ ]:
```
# evaluate on val data to inspect results
loss, accuracy, precision, recall, f1 = lstm_exp1.evaluate(X_val, y_val)
print(f'Loss: {loss}\nAccuracy: {accuracy}\nPrecision: {precision}\nRecall:␣
↪{recall},\nF1: {f1}')
```

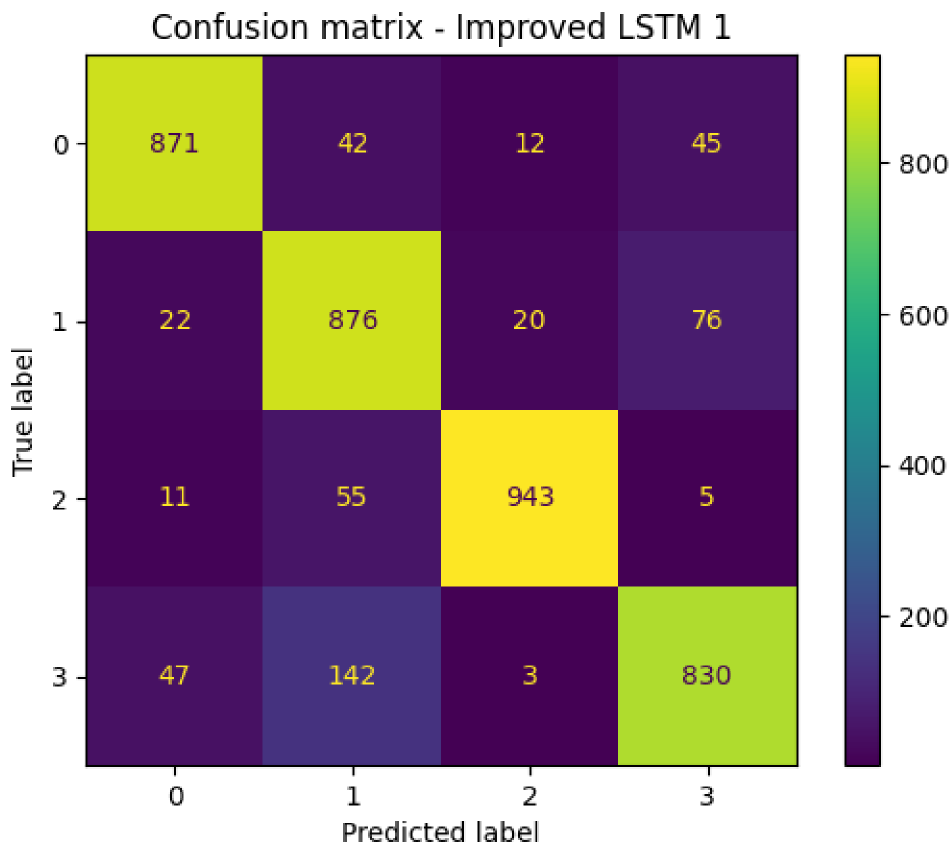
```
150/150 [==============================] - 1s 9ms/step - loss: 0.3924 -
categorical_accuracy: 0.8773 - precision_2: 0.8815 - recall_2: 0.8740 -
f1_score: 0.8776
Loss: 0.39241522550582886
Accuracy: 0.8772916793823242
Precision: 0.8814877271652222
```

```
Recall: 0.8739583492279053,
F1: [0.90870667 0.8213141  0.9464063  0.8339769 ]
```

[ ]:
```python
# plot confusion matrix
y_pred_lstm1 = lstm_exp1.predict(X_val)
cm = confusion_matrix(np.argmax(y_val, axis=1), np.argmax(y_pred_lstm1, axis=1))
ConfusionMatrixDisplay(confusion_matrix=cm).plot();
plt.title('Confusion matrix - Improved LSTM 1')
```

```
125/125 [==============================] - 2s 8ms/step
```

[ ]:
```
Text(0.5, 1.0, 'Confusion matrix - Improved LSTM 1')
```



[ ]:
```python
# free up resources
gc.collect()
```

[ ]:
```
704
```

Improved LSTM Model 2 This LSTM will add improvements to the baseline model including additional hidden LSTM layers (4) and start with the same number of units per layer (128). The units will gradually decrease with each convolutional layer down to 32. An additional fully connected layer is added. Dropout is also increased to both the convolutional layers and the fully connected layers.

[ ]:
```
# setup checkpoint
checkpoint_filepath = '/content/drive/MyDrive/USD/models/composer-classifier/
↪lstm-2'
model_checkpoint_callback = keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    monitor='val_categorical_accuracy',
    mode='max',
    save_best_only=True)

# define LSTM with mitigations for overfitting: more layers, more dropout
lstm_exp2 = tf.keras.models.Sequential([

    # input our sequence tensors
    tf.keras.layers.Input(shape=(NORM_SEQUENCE_LENGTH, NUM_PIANO_KEYS)),
    tf.keras.layers.Normalization(axis=None),
    tf.keras.layers.LSTM(128, return_sequences=True),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.LSTM(128, return_sequences=True),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.LSTM(128, return_sequences=True),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.LSTM(32, return_sequences=False),

    # classification head
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.Dense(units = NUM_COMPOSERS, activation='softmax')
])

# Compile the model
lstm_exp2.compile(
    optimizer=keras.optimizers.Adam(learning_rate=LEARNING_RATE),
    loss=tf.keras.losses.CategoricalCrossentropy(),
    metrics=[keras.metrics.CategoricalAccuracy(), keras.metrics.Precision(),␣
↪keras.metrics.Recall(), keras.metrics.F1Score()]
)
```

[ ]:
```
# Train the model
history_lstm_exp2 = lstm_exp2.fit(X_train, y_train,␣
↪validation_data=(X_val,y_val), epochs=NUM_EPOCHS, batch_size=BATCH_SIZE,␣
↪callbacks=[model_checkpoint_callback])
```

```
Epoch 1/75
1350/1350 [==============================] - 60s 39ms/step - loss: 1.1852 -
categorical_accuracy: 0.4642 - precision_4: 0.6449 - recall_4: 0.1898 -
f1_score: 0.4509 - val_loss: 1.0747 - val_categorical_accuracy: 0.5441 -
val_precision_4: 0.6349 - val_recall_4: 0.3294 - val_f1_score: 0.5324
Epoch 2/75
1350/1350 [==============================] - 50s 37ms/step - loss: 1.0040 -
categorical_accuracy: 0.5681 - precision_4: 0.6941 - recall_4: 0.3589 -
f1_score: 0.5573 - val_loss: 0.8758 - val_categorical_accuracy: 0.6380 -
val_precision_4: 0.7113 - val_recall_4: 0.4881 - val_f1_score: 0.6203
Epoch 3/75
1350/1350 [==============================] - 51s 38ms/step - loss: 0.7654 -
categorical_accuracy: 0.6838 - precision_4: 0.7375 - recall_4: 0.5942 -
f1_score: 0.6775 - val_loss: 0.6935 - val_categorical_accuracy: 0.7144 -
val_precision_4: 0.7407 - val_recall_4: 0.6702 - val_f1_score: 0.6973
Epoch 4/75
1350/1350 [==============================] - 51s 38ms/step - loss: 0.6340 -
categorical_accuracy: 0.7478 - precision_4: 0.7808 - recall_4: 0.7004 -
f1_score: 0.7432 - val_loss: 0.6165 - val_categorical_accuracy: 0.7570 -
val_precision_4: 0.7775 - val_recall_4: 0.7220 - val_f1_score: 0.7494
Epoch 5/75
1350/1350 [==============================] - 51s 37ms/step - loss: 0.5548 -
categorical_accuracy: 0.7845 - precision_4: 0.8113 - recall_4: 0.7502 -
f1_score: 0.7806 - val_loss: 0.5444 - val_categorical_accuracy: 0.7867 -
val_precision_4: 0.8140 - val_recall_4: 0.7628 - val_f1_score: 0.7761
Epoch 6/75
1350/1350 [==============================] - 51s 37ms/step - loss: 0.5024 -
categorical_accuracy: 0.8064 - precision_4: 0.8277 - recall_4: 0.7789 -
f1_score: 0.8028 - val_loss: 0.5278 - val_categorical_accuracy: 0.7978 -
val_precision_4: 0.8161 - val_recall_4: 0.7769 - val_f1_score: 0.7913
Epoch 7/75
1350/1350 [==============================] - 51s 38ms/step - loss: 0.4541 -
categorical_accuracy: 0.8275 - precision_4: 0.8462 - recall_4: 0.8036 -
f1_score: 0.8239 - val_loss: 0.5116 - val_categorical_accuracy: 0.8046 -
val_precision_4: 0.8258 - val_recall_4: 0.7841 - val_f1_score: 0.7951
Epoch 8/75
1350/1350 [==============================] - 50s 37ms/step - loss: 0.4212 -
categorical_accuracy: 0.8413 - precision_4: 0.8592 - recall_4: 0.8213 -
f1_score: 0.8380 - val_loss: 0.4875 - val_categorical_accuracy: 0.8161 -
val_precision_4: 0.8397 - val_recall_4: 0.7917 - val_f1_score: 0.8059
Epoch 9/75
1350/1350 [==============================] - 51s 38ms/step - loss: 0.3892 -
categorical_accuracy: 0.8568 - precision_4: 0.8730 - recall_4: 0.8390 -
f1_score: 0.8539 - val_loss: 0.4918 - val_categorical_accuracy: 0.8222 -
val_precision_4: 0.8354 - val_recall_4: 0.8081 - val_f1_score: 0.8171
Epoch 10/75
1350/1350 [==============================] - 49s 36ms/step - loss: 0.3615 -
categorical_accuracy: 0.8681 - precision_4: 0.8830 - recall_4: 0.8519 -
```

```
f1_score: 0.8652 - val_loss: 0.4559 - val_categorical_accuracy: 0.8359 -
val_precision_4: 0.8508 - val_recall_4: 0.8206 - val_f1_score: 0.8277
Epoch 11/75
1350/1350 [==============================] - 48s 36ms/step - loss: 0.3352 -
categorical_accuracy: 0.8793 - precision_4: 0.8928 - recall_4: 0.8654 -
f1_score: 0.8767 - val_loss: 0.4021 - val_categorical_accuracy: 0.8504 -
val_precision_4: 0.8614 - val_recall_4: 0.8400 - val_f1_score: 0.8445
Epoch 12/75
1350/1350 [==============================] - 48s 36ms/step - loss: 0.3220 -
categorical_accuracy: 0.8844 - precision_4: 0.8960 - recall_4: 0.8715 -
f1_score: 0.8818 - val_loss: 0.4065 - val_categorical_accuracy: 0.8515 -
val_precision_4: 0.8624 - val_recall_4: 0.8411 - val_f1_score: 0.8470
Epoch 13/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.2990 -
categorical_accuracy: 0.8906 - precision_4: 0.9034 - recall_4: 0.8785 -
f1_score: 0.8882 - val_loss: 0.4341 - val_categorical_accuracy: 0.8406 -
val_precision_4: 0.8539 - val_recall_4: 0.8300 - val_f1_score: 0.8328
Epoch 14/75
1350/1350 [==============================] - 48s 35ms/step - loss: 0.2850 -
categorical_accuracy: 0.8976 - precision_4: 0.9085 - recall_4: 0.8863 -
f1_score: 0.8955 - val_loss: 0.4352 - val_categorical_accuracy: 0.8522 -
val_precision_4: 0.8635 - val_recall_4: 0.8389 - val_f1_score: 0.8505
Epoch 15/75
1350/1350 [==============================] - 48s 36ms/step - loss: 0.2664 -
categorical_accuracy: 0.9046 - precision_4: 0.9146 - recall_4: 0.8941 -
f1_score: 0.9024 - val_loss: 0.3801 - val_categorical_accuracy: 0.8643 -
val_precision_4: 0.8763 - val_recall_4: 0.8524 - val_f1_score: 0.8593
Epoch 16/75
1350/1350 [==============================] - 48s 36ms/step - loss: 0.2482 -
categorical_accuracy: 0.9144 - precision_4: 0.9215 - recall_4: 0.9059 -
f1_score: 0.9124 - val_loss: 0.3693 - val_categorical_accuracy: 0.8674 -
val_precision_4: 0.8774 - val_recall_4: 0.8602 - val_f1_score: 0.8633
Epoch 17/75
1350/1350 [==============================] - 37s 28ms/step - loss: 0.2425 -
categorical_accuracy: 0.9153 - precision_4: 0.9234 - recall_4: 0.9069 -
f1_score: 0.9134 - val_loss: 0.4199 - val_categorical_accuracy: 0.8609 -
val_precision_4: 0.8710 - val_recall_4: 0.8519 - val_f1_score: 0.8570
Epoch 18/75
1350/1350 [==============================] - 48s 36ms/step - loss: 0.2297 -
categorical_accuracy: 0.9190 - precision_4: 0.9266 - recall_4: 0.9113 -
f1_score: 0.9174 - val_loss: 0.3873 - val_categorical_accuracy: 0.8707 -
val_precision_4: 0.8773 - val_recall_4: 0.8661 - val_f1_score: 0.8660
Epoch 19/75
1350/1350 [==============================] - 47s 35ms/step - loss: 0.2246 -
categorical_accuracy: 0.9208 - precision_4: 0.9280 - recall_4: 0.9135 -
f1_score: 0.9191 - val_loss: 0.3785 - val_categorical_accuracy: 0.8724 -
val_precision_4: 0.8781 - val_recall_4: 0.8617 - val_f1_score: 0.8694
Epoch 20/75
```

```
1350/1350 [==============================] - 48s 36ms/step - loss: 0.2088 -
categorical_accuracy: 0.9258 - precision_4: 0.9332 - recall_4: 0.9196 -
f1_score: 0.9242 - val_loss: 0.3777 - val_categorical_accuracy: 0.8785 -
val_precision_4: 0.8859 - val_recall_4: 0.8728 - val_f1_score: 0.8746
Epoch 21/75
1350/1350 [==============================] - 48s 36ms/step - loss: 0.2022 -
categorical_accuracy: 0.9300 - precision_4: 0.9366 - recall_4: 0.9231 -
f1_score: 0.9285 - val_loss: 0.3698 - val_categorical_accuracy: 0.8796 -
val_precision_4: 0.8867 - val_recall_4: 0.8728 - val_f1_score: 0.8754
Epoch 22/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.2062 -
categorical_accuracy: 0.9279 - precision_4: 0.9345 - recall_4: 0.9215 -
f1_score: 0.9265 - val_loss: 0.3698 - val_categorical_accuracy: 0.8772 -
val_precision_4: 0.8837 - val_recall_4: 0.8698 - val_f1_score: 0.8716
Epoch 23/75
1350/1350 [==============================] - 48s 35ms/step - loss: 0.1971 -
categorical_accuracy: 0.9324 - precision_4: 0.9380 - recall_4: 0.9266 -
f1_score: 0.9309 - val_loss: 0.3619 - val_categorical_accuracy: 0.8874 -
val_precision_4: 0.8940 - val_recall_4: 0.8828 - val_f1_score: 0.8835
Epoch 24/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1867 -
categorical_accuracy: 0.9339 - precision_4: 0.9404 - recall_4: 0.9286 -
f1_score: 0.9325 - val_loss: 0.3847 - val_categorical_accuracy: 0.8698 -
val_precision_4: 0.8809 - val_recall_4: 0.8615 - val_f1_score: 0.8631
Epoch 25/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1786 -
categorical_accuracy: 0.9375 - precision_4: 0.9431 - recall_4: 0.9316 -
f1_score: 0.9361 - val_loss: 0.3657 - val_categorical_accuracy: 0.8843 -
val_precision_4: 0.8913 - val_recall_4: 0.8807 - val_f1_score: 0.8802
Epoch 26/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1722 -
categorical_accuracy: 0.9404 - precision_4: 0.9449 - recall_4: 0.9361 -
f1_score: 0.9390 - val_loss: 0.3790 - val_categorical_accuracy: 0.8789 -
val_precision_4: 0.8843 - val_recall_4: 0.8748 - val_f1_score: 0.8753
Epoch 27/75
1350/1350 [==============================] - 48s 35ms/step - loss: 0.1630 -
categorical_accuracy: 0.9434 - precision_4: 0.9476 - recall_4: 0.9394 -
f1_score: 0.9421 - val_loss: 0.3622 - val_categorical_accuracy: 0.8943 -
val_precision_4: 0.9006 - val_recall_4: 0.8906 - val_f1_score: 0.8906
Epoch 28/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1568 -
categorical_accuracy: 0.9446 - precision_4: 0.9493 - recall_4: 0.9406 -
f1_score: 0.9434 - val_loss: 0.4232 - val_categorical_accuracy: 0.8824 -
val_precision_4: 0.8849 - val_recall_4: 0.8783 - val_f1_score: 0.8799
Epoch 29/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1604 -
categorical_accuracy: 0.9444 - precision_4: 0.9487 - recall_4: 0.9403 -
f1_score: 0.9431 - val_loss: 0.3599 - val_categorical_accuracy: 0.8856 -
```

```
val_precision_4: 0.8902 - val_recall_4: 0.8817 - val_f1_score: 0.8811
Epoch 30/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1497 -
categorical_accuracy: 0.9482 - precision_4: 0.9523 - recall_4: 0.9440 -
f1_score: 0.9471 - val_loss: 0.3502 - val_categorical_accuracy: 0.8843 -
val_precision_4: 0.8927 - val_recall_4: 0.8794 - val_f1_score: 0.8795
Epoch 31/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1466 -
categorical_accuracy: 0.9491 - precision_4: 0.9535 - recall_4: 0.9452 -
f1_score: 0.9480 - val_loss: 0.3409 - val_categorical_accuracy: 0.8920 -
val_precision_4: 0.8973 - val_recall_4: 0.8885 - val_f1_score: 0.8882
Epoch 32/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1465 -
categorical_accuracy: 0.9496 - precision_4: 0.9531 - recall_4: 0.9464 -
f1_score: 0.9485 - val_loss: 0.3772 - val_categorical_accuracy: 0.8870 -
val_precision_4: 0.8934 - val_recall_4: 0.8835 - val_f1_score: 0.8831
Epoch 33/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1391 -
categorical_accuracy: 0.9505 - precision_4: 0.9542 - recall_4: 0.9473 -
f1_score: 0.9494 - val_loss: 0.3880 - val_categorical_accuracy: 0.8937 -
val_precision_4: 0.8987 - val_recall_4: 0.8902 - val_f1_score: 0.8896
Epoch 34/75
1350/1350 [==============================] - 48s 35ms/step - loss: 0.1327 -
categorical_accuracy: 0.9553 - precision_4: 0.9586 - recall_4: 0.9516 -
f1_score: 0.9543 - val_loss: 0.3311 - val_categorical_accuracy: 0.9007 -
val_precision_4: 0.9053 - val_recall_4: 0.8963 - val_f1_score: 0.8976
Epoch 35/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1317 -
categorical_accuracy: 0.9546 - precision_4: 0.9583 - recall_4: 0.9510 -
f1_score: 0.9537 - val_loss: 0.3681 - val_categorical_accuracy: 0.8959 -
val_precision_4: 0.9003 - val_recall_4: 0.8928 - val_f1_score: 0.8925
Epoch 36/75
1350/1350 [==============================] - 36s 27ms/step - loss: 0.1310 -
categorical_accuracy: 0.9532 - precision_4: 0.9569 - recall_4: 0.9503 -
f1_score: 0.9521 - val_loss: 0.3789 - val_categorical_accuracy: 0.8998 -
val_precision_4: 0.9030 - val_recall_4: 0.8963 - val_f1_score: 0.8960
Epoch 37/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1280 -
categorical_accuracy: 0.9548 - precision_4: 0.9583 - recall_4: 0.9517 -
f1_score: 0.9538 - val_loss: 0.3814 - val_categorical_accuracy: 0.8931 -
val_precision_4: 0.8981 - val_recall_4: 0.8900 - val_f1_score: 0.8890
Epoch 38/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1275 -
categorical_accuracy: 0.9550 - precision_4: 0.9580 - recall_4: 0.9526 -
f1_score: 0.9539 - val_loss: 0.3708 - val_categorical_accuracy: 0.8924 -
val_precision_4: 0.8967 - val_recall_4: 0.8872 - val_f1_score: 0.8884
Epoch 39/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1242 -
```

```
categorical_accuracy: 0.9577 - precision_4: 0.9607 - recall_4: 0.9546 -
f1_score: 0.9569 - val_loss: 0.3584 - val_categorical_accuracy: 0.8985 -
val_precision_4: 0.9027 - val_recall_4: 0.8948 - val_f1_score: 0.8945
Epoch 40/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1189 -
categorical_accuracy: 0.9584 - precision_4: 0.9609 - recall_4: 0.9557 -
f1_score: 0.9575 - val_loss: 0.3700 - val_categorical_accuracy: 0.8930 -
val_precision_4: 0.8958 - val_recall_4: 0.8919 - val_f1_score: 0.8892
Epoch 41/75
1350/1350 [==============================] - 36s 27ms/step - loss: 0.1160 -
categorical_accuracy: 0.9600 - precision_4: 0.9624 - recall_4: 0.9576 -
f1_score: 0.9593 - val_loss: 0.3526 - val_categorical_accuracy: 0.9007 -
val_precision_4: 0.9045 - val_recall_4: 0.8980 - val_f1_score: 0.8966
Epoch 42/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1115 -
categorical_accuracy: 0.9620 - precision_4: 0.9641 - recall_4: 0.9599 -
f1_score: 0.9611 - val_loss: 0.3609 - val_categorical_accuracy: 0.8978 -
val_precision_4: 0.9012 - val_recall_4: 0.8957 - val_f1_score: 0.8942
Epoch 43/75
1350/1350 [==============================] - 48s 35ms/step - loss: 0.1123 -
categorical_accuracy: 0.9621 - precision_4: 0.9646 - recall_4: 0.9600 -
f1_score: 0.9614 - val_loss: 0.3289 - val_categorical_accuracy: 0.9017 -
val_precision_4: 0.9058 - val_recall_4: 0.8989 - val_f1_score: 0.8981
Epoch 44/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1142 -
categorical_accuracy: 0.9606 - precision_4: 0.9631 - recall_4: 0.9579 -
f1_score: 0.9597 - val_loss: 0.3298 - val_categorical_accuracy: 0.9007 -
val_precision_4: 0.9040 - val_recall_4: 0.8981 - val_f1_score: 0.8968
Epoch 45/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1098 -
categorical_accuracy: 0.9624 - precision_4: 0.9647 - recall_4: 0.9603 -
f1_score: 0.9616 - val_loss: 0.3427 - val_categorical_accuracy: 0.8978 -
val_precision_4: 0.9017 - val_recall_4: 0.8954 - val_f1_score: 0.8943
Epoch 46/75
1350/1350 [==============================] - 48s 35ms/step - loss: 0.1065 -
categorical_accuracy: 0.9642 - precision_4: 0.9662 - recall_4: 0.9619 -
f1_score: 0.9634 - val_loss: 0.3517 - val_categorical_accuracy: 0.9065 -
val_precision_4: 0.9088 - val_recall_4: 0.9044 - val_f1_score: 0.9028
Epoch 47/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1078 -
categorical_accuracy: 0.9642 - precision_4: 0.9667 - recall_4: 0.9622 -
f1_score: 0.9635 - val_loss: 0.3465 - val_categorical_accuracy: 0.9030 -
val_precision_4: 0.9054 - val_recall_4: 0.9007 - val_f1_score: 0.8996
Epoch 48/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1081 -
categorical_accuracy: 0.9617 - precision_4: 0.9636 - recall_4: 0.9597 -
f1_score: 0.9608 - val_loss: 0.3594 - val_categorical_accuracy: 0.8985 -
val_precision_4: 0.9041 - val_recall_4: 0.8952 - val_f1_score: 0.8946
```

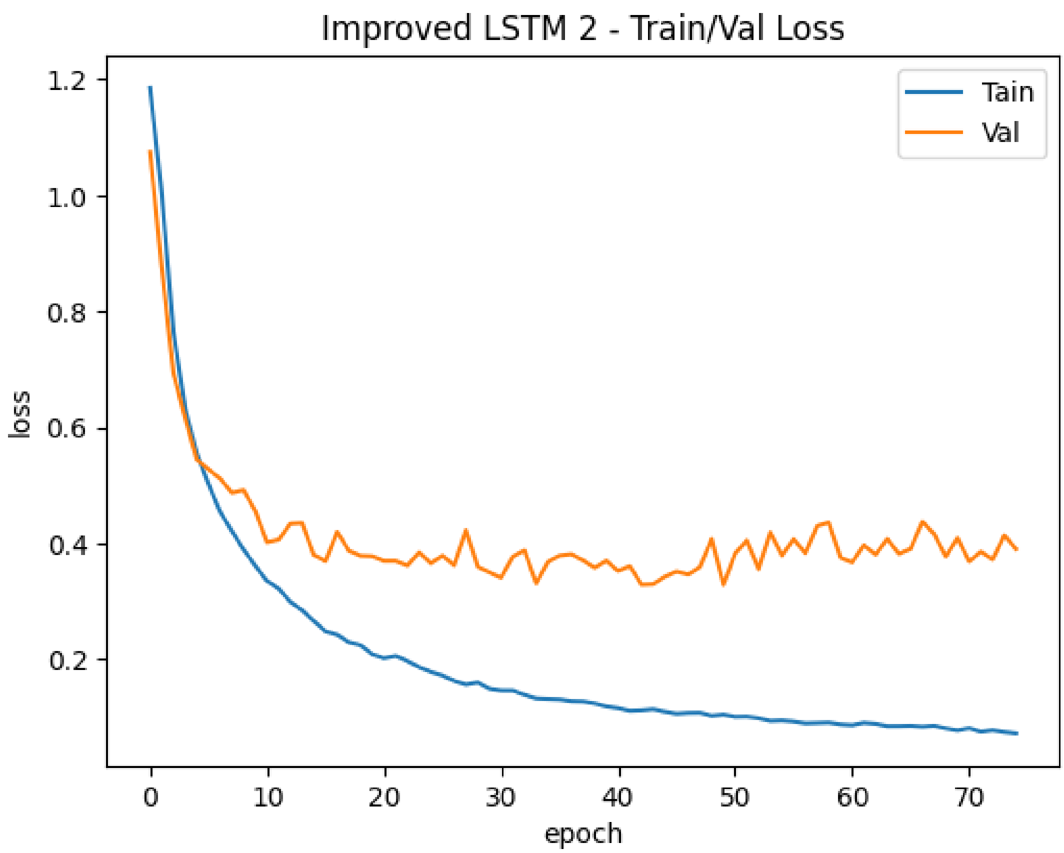
```
Epoch 49/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1029 -
categorical_accuracy: 0.9648 - precision_4: 0.9667 - recall_4: 0.9627 -
f1_score: 0.9641 - val_loss: 0.4078 - val_categorical_accuracy: 0.8926 -
val_precision_4: 0.8950 - val_recall_4: 0.8915 - val_f1_score: 0.8871
Epoch 50/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1050 -
categorical_accuracy: 0.9637 - precision_4: 0.9657 - recall_4: 0.9618 -
f1_score: 0.9628 - val_loss: 0.3286 - val_categorical_accuracy: 0.9017 -
val_precision_4: 0.9056 - val_recall_4: 0.8985 - val_f1_score: 0.8979
Epoch 51/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.1011 -
categorical_accuracy: 0.9655 - precision_4: 0.9672 - recall_4: 0.9641 -
f1_score: 0.9647 - val_loss: 0.3826 - val_categorical_accuracy: 0.8965 -
val_precision_4: 0.9000 - val_recall_4: 0.8937 - val_f1_score: 0.8929
Epoch 52/75
1350/1350 [==============================] - 36s 27ms/step - loss: 0.1015 -
categorical_accuracy: 0.9660 - precision_4: 0.9680 - recall_4: 0.9643 -
f1_score: 0.9652 - val_loss: 0.4052 - val_categorical_accuracy: 0.8985 -
val_precision_4: 0.9012 - val_recall_4: 0.8974 - val_f1_score: 0.8947
Epoch 53/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0988 -
categorical_accuracy: 0.9673 - precision_4: 0.9694 - recall_4: 0.9657 -
f1_score: 0.9665 - val_loss: 0.3554 - val_categorical_accuracy: 0.9019 -
val_precision_4: 0.9040 - val_recall_4: 0.8994 - val_f1_score: 0.8986
Epoch 54/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0941 -
categorical_accuracy: 0.9683 - precision_4: 0.9698 - recall_4: 0.9672 -
f1_score: 0.9677 - val_loss: 0.4195 - val_categorical_accuracy: 0.8946 -
val_precision_4: 0.8961 - val_recall_4: 0.8931 - val_f1_score: 0.8911
Epoch 55/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0948 -
categorical_accuracy: 0.9680 - precision_4: 0.9699 - recall_4: 0.9663 -
f1_score: 0.9674 - val_loss: 0.3788 - val_categorical_accuracy: 0.9009 -
val_precision_4: 0.9029 - val_recall_4: 0.8985 - val_f1_score: 0.8974
Epoch 56/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0930 -
categorical_accuracy: 0.9682 - precision_4: 0.9698 - recall_4: 0.9666 -
f1_score: 0.9676 - val_loss: 0.4077 - val_categorical_accuracy: 0.9031 -
val_precision_4: 0.9047 - val_recall_4: 0.9020 - val_f1_score: 0.8999
Epoch 57/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0899 -
categorical_accuracy: 0.9691 - precision_4: 0.9706 - recall_4: 0.9676 -
f1_score: 0.9685 - val_loss: 0.3829 - val_categorical_accuracy: 0.9028 -
val_precision_4: 0.9057 - val_recall_4: 0.9004 - val_f1_score: 0.8990
Epoch 58/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0905 -
categorical_accuracy: 0.9689 - precision_4: 0.9711 - recall_4: 0.9675 -
```

```
f1_score: 0.9682 - val_loss: 0.4299 - val_categorical_accuracy: 0.9002 -
val_precision_4: 0.9025 - val_recall_4: 0.8983 - val_f1_score: 0.8961
Epoch 59/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0912 -
categorical_accuracy: 0.9693 - precision_4: 0.9709 - recall_4: 0.9677 -
f1_score: 0.9686 - val_loss: 0.4362 - val_categorical_accuracy: 0.8956 -
val_precision_4: 0.8986 - val_recall_4: 0.8946 - val_f1_score: 0.8914
Epoch 60/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0880 -
categorical_accuracy: 0.9701 - precision_4: 0.9720 - recall_4: 0.9684 -
f1_score: 0.9695 - val_loss: 0.3754 - val_categorical_accuracy: 0.9041 -
val_precision_4: 0.9058 - val_recall_4: 0.9013 - val_f1_score: 0.9005
Epoch 61/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0867 -
categorical_accuracy: 0.9694 - precision_4: 0.9710 - recall_4: 0.9682 -
f1_score: 0.9687 - val_loss: 0.3674 - val_categorical_accuracy: 0.9015 -
val_precision_4: 0.9039 - val_recall_4: 0.8993 - val_f1_score: 0.8975
Epoch 62/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0907 -
categorical_accuracy: 0.9694 - precision_4: 0.9709 - recall_4: 0.9678 -
f1_score: 0.9688 - val_loss: 0.3971 - val_categorical_accuracy: 0.9015 -
val_precision_4: 0.9033 - val_recall_4: 0.9000 - val_f1_score: 0.8982
Epoch 63/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0890 -
categorical_accuracy: 0.9690 - precision_4: 0.9707 - recall_4: 0.9671 -
f1_score: 0.9683 - val_loss: 0.3807 - val_categorical_accuracy: 0.9013 -
val_precision_4: 0.9039 - val_recall_4: 0.9007 - val_f1_score: 0.8975
Epoch 64/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0849 -
categorical_accuracy: 0.9707 - precision_4: 0.9721 - recall_4: 0.9692 -
f1_score: 0.9701 - val_loss: 0.4078 - val_categorical_accuracy: 0.8970 -
val_precision_4: 0.8992 - val_recall_4: 0.8952 - val_f1_score: 0.8934
Epoch 65/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0850 -
categorical_accuracy: 0.9718 - precision_4: 0.9732 - recall_4: 0.9703 -
f1_score: 0.9712 - val_loss: 0.3820 - val_categorical_accuracy: 0.9019 -
val_precision_4: 0.9052 - val_recall_4: 0.9002 - val_f1_score: 0.8984
Epoch 66/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0855 -
categorical_accuracy: 0.9713 - precision_4: 0.9729 - recall_4: 0.9700 -
f1_score: 0.9707 - val_loss: 0.3907 - val_categorical_accuracy: 0.9039 -
val_precision_4: 0.9059 - val_recall_4: 0.9019 - val_f1_score: 0.9004
Epoch 67/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0841 -
categorical_accuracy: 0.9714 - precision_4: 0.9731 - recall_4: 0.9700 -
f1_score: 0.9707 - val_loss: 0.4374 - val_categorical_accuracy: 0.9007 -
val_precision_4: 0.9035 - val_recall_4: 0.8994 - val_f1_score: 0.8975
Epoch 68/75
```

```
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0855 -
categorical_accuracy: 0.9710 - precision_4: 0.9725 - recall_4: 0.9693 -
f1_score: 0.9704 - val_loss: 0.4165 - val_categorical_accuracy: 0.9020 -
val_precision_4: 0.9042 - val_recall_4: 0.9004 - val_f1_score: 0.8985
Epoch 69/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0812 -
categorical_accuracy: 0.9725 - precision_4: 0.9738 - recall_4: 0.9712 -
f1_score: 0.9720 - val_loss: 0.3774 - val_categorical_accuracy: 0.9013 -
val_precision_4: 0.9031 - val_recall_4: 0.9006 - val_f1_score: 0.8980
Epoch 70/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0778 -
categorical_accuracy: 0.9748 - precision_4: 0.9759 - recall_4: 0.9736 -
f1_score: 0.9743 - val_loss: 0.4091 - val_categorical_accuracy: 0.9019 -
val_precision_4: 0.9036 - val_recall_4: 0.8996 - val_f1_score: 0.8984
Epoch 71/75
1350/1350 [==============================] - 48s 36ms/step - loss: 0.0815 -
categorical_accuracy: 0.9712 - precision_4: 0.9729 - recall_4: 0.9695 -
f1_score: 0.9707 - val_loss: 0.3689 - val_categorical_accuracy: 0.9074 -
val_precision_4: 0.9093 - val_recall_4: 0.9059 - val_f1_score: 0.9038
Epoch 72/75
1350/1350 [==============================] - 37s 28ms/step - loss: 0.0754 -
categorical_accuracy: 0.9738 - precision_4: 0.9748 - recall_4: 0.9730 -
f1_score: 0.9732 - val_loss: 0.3856 - val_categorical_accuracy: 0.9043 -
val_precision_4: 0.9061 - val_recall_4: 0.9024 - val_f1_score: 0.9003
Epoch 73/75
1350/1350 [==============================] - 48s 35ms/step - loss: 0.0780 -
categorical_accuracy: 0.9735 - precision_4: 0.9747 - recall_4: 0.9723 -
f1_score: 0.9729 - val_loss: 0.3723 - val_categorical_accuracy: 0.9091 -
val_precision_4: 0.9114 - val_recall_4: 0.9085 - val_f1_score: 0.9056
Epoch 74/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0748 -
categorical_accuracy: 0.9746 - precision_4: 0.9759 - recall_4: 0.9735 -
f1_score: 0.9741 - val_loss: 0.4140 - val_categorical_accuracy: 0.9033 -
val_precision_4: 0.9050 - val_recall_4: 0.9015 - val_f1_score: 0.8996
Epoch 75/75
1350/1350 [==============================] - 37s 27ms/step - loss: 0.0725 -
categorical_accuracy: 0.9756 - precision_4: 0.9766 - recall_4: 0.9746 -
f1_score: 0.9752 - val_loss: 0.3904 - val_categorical_accuracy: 0.9083 -
val_precision_4: 0.9105 - val_recall_4: 0.9065 - val_f1_score: 0.9050
```

[ ]:
```
#plot loss
plt.plot(history_lstm_exp2.history['loss'])
plt.plot(history_lstm_exp2.history['val_loss'])
plt.title('Improved LSTM 2 - Train/Val Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Tain', 'Val'])
```

```
plt.show()
```



[ ]:
```
gc.collect()
```

[ ]: 154176

[ ]:
```
# load best model from checkpoint
lstm_exp2_best = tf.keras.models.load_model('/content/drive/MyDrive/USD/models/
 ↪composer-classifier/lstm-2')
```

[ ]:
```
# evaluate on test data
loss, accuracy, precision, recall, f1 = lstm_exp2_best.evaluate(X_val, y_val)
print(f'Loss: {loss}\nAccuracy: {accuracy}\nPrecision: {precision}\nRecall:␣
 ↪{recall},\nF1: {f1}')
```

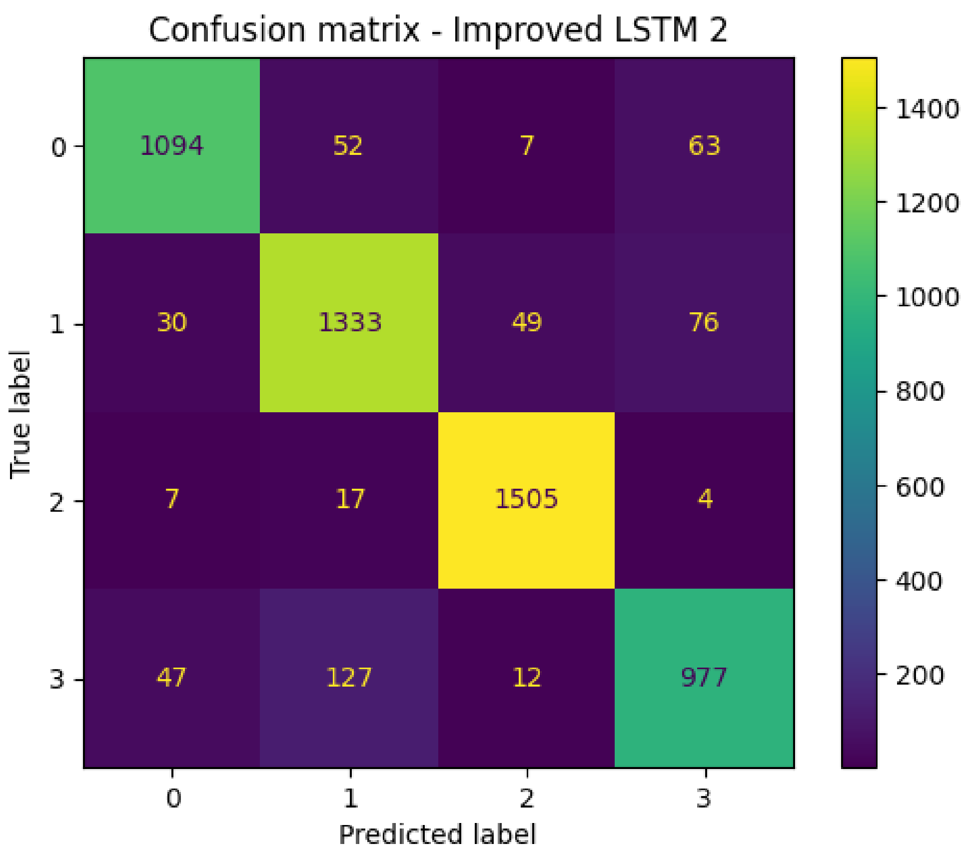
```
169/169 [==============================] - 5s 14ms/step - loss: 0.3723 -
categorical_accuracy: 0.9091 - precision_4: 0.9114 - recall_4: 0.9085 -
f1_score: 0.9056
Loss: 0.3723032772541046
Accuracy: 0.909074068069458
```

```
Precision: 0.9113876819610596
Recall: 0.9085184931755066,
F1: [0.9139515  0.88365924 0.9690921  0.8558914 ]
```

[ ]:
```
# plot confusion matrix
y_pred_lstm2 = lstm_exp2_best.predict(X_val)
cm = confusion_matrix(np.argmax(y_val, axis=1), np.argmax(y_pred_lstm2, axis=1))
ConfusionMatrixDisplay(confusion_matrix=cm).plot();
plt.title('Confusion matrix - Improved LSTM 2')
```

```
169/169 [==============================] - 3s 12ms/step
```

[ ]:
```
Text(0.5, 1.0, 'Confusion matrix - Improved LSTM 2')
```



1.4.2 CNN Models

For next set of models, we will define and train CNN-based models to process our sequences and perform a classification task to predict the appropriate composer. Some experimenation and fine tuning will be conducted to find an optimal model definition.

1. Define baseline CNN model with classification output layer. This will be used to validate our processed data, validate classification task and set baseline performance.

2. Train model on our training set
3. Evaluate performance of the model using Accuracy, Precision/Recall, F1
4. Tune hyperparameters and model architecture

[ ]:
```
# free up resources
gc.collect()
```

[ ]: 9547

Baseline CNN Model This CNN is not as simple as the baseline LSTM, but is still relatively simple CNN with 3 convolutional layers and no dropout or other regularization techniques.

[ ]:
```
# setup checkpoint
checkpoint_filepath = '/content/drive/MyDrive/USD/models/composer-classifier/
↪cnn-1'
model_checkpoint_callback = keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    monitor='val_categorical_accuracy',
    mode='max',
    save_best_only=True)

cnn_baseline = tf.keras.Sequential([

  #tf.keras.layers.Input(shape=(NORM_SEQUENCE_LENGTH, NUM_PIANO_KEYS)),
  tf.keras.layers.Normalization(axis=None),

  tf.keras.layers.Conv1D(256, kernel_size=3, activation='relu',␣
↪dilation_rate=2, padding='causal'),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Conv1D(128, kernel_size=3, activation='relu',␣
↪dilation_rate=2, padding='causal'),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Conv1D(64, kernel_size=3, activation='relu', dilation_rate=2,␣
↪padding='causal'),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Flatten(),

  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(NUM_COMPOSERS, activation='softmax')
])
```

```
# Compile the model
cnn_baseline.compile(
    optimizer=keras.optimizers.Adam(learning_rate=LEARNING_RATE),
    loss=tf.keras.losses.CategoricalCrossentropy(),
    metrics=[keras.metrics.CategoricalAccuracy(), keras.metrics.Precision(),␣
 ↪keras.metrics.Recall(), keras.metrics.F1Score(),␣
 ↪callbacks=[model_checkpoint_callback]]
)
```

[ ]:
```
# Train the model
history_cnn_baseline = cnn_baseline.fit(X_train, y_train,␣
 ↪validation_data=(X_val,y_val), epochs=NUM_EPOCHS, batch_size=BATCH_SIZE)
```

```
Epoch 1/75
1200/1200 [==============================] - 17s 8ms/step - loss: 1.2457 -
categorical_accuracy: 0.4482 - precision_4: 0.6511 - recall_4: 0.1861 -
f1_score: 0.4429 - val_loss: 0.9901 - val_categorical_accuracy: 0.5783 -
val_precision_4: 0.7135 - val_recall_4: 0.3990 - val_f1_score: 0.5422
Epoch 2/75
1200/1200 [==============================] - 9s 7ms/step - loss: 0.8529 -
categorical_accuracy: 0.6524 - precision_4: 0.7427 - recall_4: 0.4992 -
f1_score: 0.6426 - val_loss: 0.7548 - val_categorical_accuracy: 0.6929 -
val_precision_4: 0.7861 - val_recall_4: 0.5404 - val_f1_score: 0.6787
Epoch 3/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.6896 -
categorical_accuracy: 0.7208 - precision_4: 0.7830 - recall_4: 0.6328 -
f1_score: 0.7149 - val_loss: 0.6618 - val_categorical_accuracy: 0.7340 -
val_precision_4: 0.8012 - val_recall_4: 0.6379 - val_f1_score: 0.7217
Epoch 4/75
1200/1200 [==============================] - 9s 7ms/step - loss: 0.5888 -
categorical_accuracy: 0.7633 - precision_4: 0.8097 - recall_4: 0.7020 -
f1_score: 0.7602 - val_loss: 0.6283 - val_categorical_accuracy: 0.7563 -
val_precision_4: 0.7997 - val_recall_4: 0.7146 - val_f1_score: 0.7528
Epoch 5/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.5230 -
categorical_accuracy: 0.7942 - precision_4: 0.8274 - recall_4: 0.7519 -
f1_score: 0.7921 - val_loss: 0.5736 - val_categorical_accuracy: 0.7738 -
val_precision_4: 0.8129 - val_recall_4: 0.7258 - val_f1_score: 0.7668
Epoch 6/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.4638 -
categorical_accuracy: 0.8199 - precision_4: 0.8467 - recall_4: 0.7848 -
f1_score: 0.8187 - val_loss: 0.5770 - val_categorical_accuracy: 0.7833 -
val_precision_4: 0.8107 - val_recall_4: 0.7477 - val_f1_score: 0.7785
Epoch 7/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.4191 -
categorical_accuracy: 0.8396 - precision_4: 0.8643 - recall_4: 0.8108 -
f1_score: 0.8388 - val_loss: 0.5031 - val_categorical_accuracy: 0.8133 -
val_precision_4: 0.8431 - val_recall_4: 0.7767 - val_f1_score: 0.8099
```

```
Epoch 8/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.3840 -
categorical_accuracy: 0.8557 - precision_4: 0.8747 - recall_4: 0.8333 -
f1_score: 0.8550 - val_loss: 0.4990 - val_categorical_accuracy: 0.8167 -
val_precision_4: 0.8376 - val_recall_4: 0.7921 - val_f1_score: 0.8150
Epoch 9/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.3446 -
categorical_accuracy: 0.8719 - precision_4: 0.8888 - recall_4: 0.8518 -
f1_score: 0.8714 - val_loss: 0.5649 - val_categorical_accuracy: 0.8179 -
val_precision_4: 0.8326 - val_recall_4: 0.7981 - val_f1_score: 0.8174
Epoch 10/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.3040 -
categorical_accuracy: 0.8890 - precision_4: 0.9020 - recall_4: 0.8743 -
f1_score: 0.8887 - val_loss: 0.4968 - val_categorical_accuracy: 0.8383 -
val_precision_4: 0.8547 - val_recall_4: 0.8188 - val_f1_score: 0.8348
Epoch 11/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.2864 -
categorical_accuracy: 0.8960 - precision_4: 0.9071 - recall_4: 0.8827 -
f1_score: 0.8958 - val_loss: 0.4942 - val_categorical_accuracy: 0.8250 -
val_precision_4: 0.8445 - val_recall_4: 0.8065 - val_f1_score: 0.8261
Epoch 12/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.2535 -
categorical_accuracy: 0.9099 - precision_4: 0.9212 - recall_4: 0.8990 -
f1_score: 0.9096 - val_loss: 0.5004 - val_categorical_accuracy: 0.8452 -
val_precision_4: 0.8578 - val_recall_4: 0.8310 - val_f1_score: 0.8437
Epoch 13/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.2427 -
categorical_accuracy: 0.9145 - precision_4: 0.9244 - recall_4: 0.9036 -
f1_score: 0.9144 - val_loss: 0.4824 - val_categorical_accuracy: 0.8462 -
val_precision_4: 0.8590 - val_recall_4: 0.8354 - val_f1_score: 0.8458
Epoch 14/75
1200/1200 [==============================] - 9s 7ms/step - loss: 0.2110 -
categorical_accuracy: 0.9274 - precision_4: 0.9350 - recall_4: 0.9198 -
f1_score: 0.9274 - val_loss: 0.5326 - val_categorical_accuracy: 0.8467 -
val_precision_4: 0.8579 - val_recall_4: 0.8365 - val_f1_score: 0.8461
Epoch 15/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1889 -
categorical_accuracy: 0.9334 - precision_4: 0.9398 - recall_4: 0.9267 -
f1_score: 0.9333 - val_loss: 0.5744 - val_categorical_accuracy: 0.8483 -
val_precision_4: 0.8587 - val_recall_4: 0.8421 - val_f1_score: 0.8472
Epoch 16/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1955 -
categorical_accuracy: 0.9342 - precision_4: 0.9416 - recall_4: 0.9267 -
f1_score: 0.9342 - val_loss: 0.4730 - val_categorical_accuracy: 0.8587 -
val_precision_4: 0.8707 - val_recall_4: 0.8471 - val_f1_score: 0.8580
Epoch 17/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1708 -
categorical_accuracy: 0.9437 - precision_4: 0.9491 - recall_4: 0.9381 -
```

```
f1_score: 0.9437 - val_loss: 0.5324 - val_categorical_accuracy: 0.8512 -
val_precision_4: 0.8588 - val_recall_4: 0.8442 - val_f1_score: 0.8519
Epoch 18/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1656 -
categorical_accuracy: 0.9440 - precision_4: 0.9497 - recall_4: 0.9389 -
f1_score: 0.9440 - val_loss: 0.5318 - val_categorical_accuracy: 0.8448 -
val_precision_4: 0.8578 - val_recall_4: 0.8331 - val_f1_score: 0.8449
Epoch 19/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1599 -
categorical_accuracy: 0.9467 - precision_4: 0.9524 - recall_4: 0.9413 -
f1_score: 0.9467 - val_loss: 0.5388 - val_categorical_accuracy: 0.8469 -
val_precision_4: 0.8593 - val_recall_4: 0.8388 - val_f1_score: 0.8480
Epoch 20/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1462 -
categorical_accuracy: 0.9521 - precision_4: 0.9565 - recall_4: 0.9477 -
f1_score: 0.9521 - val_loss: 0.5671 - val_categorical_accuracy: 0.8544 -
val_precision_4: 0.8621 - val_recall_4: 0.8494 - val_f1_score: 0.8540
Epoch 21/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1417 -
categorical_accuracy: 0.9555 - precision_4: 0.9589 - recall_4: 0.9515 -
f1_score: 0.9555 - val_loss: 0.6164 - val_categorical_accuracy: 0.8644 -
val_precision_4: 0.8700 - val_recall_4: 0.8577 - val_f1_score: 0.8652
Epoch 22/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1447 -
categorical_accuracy: 0.9540 - precision_4: 0.9588 - recall_4: 0.9488 -
f1_score: 0.9540 - val_loss: 0.4865 - val_categorical_accuracy: 0.8617 -
val_precision_4: 0.8708 - val_recall_4: 0.8535 - val_f1_score: 0.8608
Epoch 23/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1258 -
categorical_accuracy: 0.9611 - precision_4: 0.9645 - recall_4: 0.9578 -
f1_score: 0.9612 - val_loss: 0.5736 - val_categorical_accuracy: 0.8673 -
val_precision_4: 0.8706 - val_recall_4: 0.8631 - val_f1_score: 0.8671
Epoch 24/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1227 -
categorical_accuracy: 0.9613 - precision_4: 0.9641 - recall_4: 0.9579 -
f1_score: 0.9613 - val_loss: 0.5752 - val_categorical_accuracy: 0.8677 -
val_precision_4: 0.8751 - val_recall_4: 0.8644 - val_f1_score: 0.8671
Epoch 25/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1218 -
categorical_accuracy: 0.9630 - precision_4: 0.9659 - recall_4: 0.9595 -
f1_score: 0.9630 - val_loss: 0.5913 - val_categorical_accuracy: 0.8606 -
val_precision_4: 0.8685 - val_recall_4: 0.8533 - val_f1_score: 0.8604
Epoch 26/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1163 -
categorical_accuracy: 0.9653 - precision_4: 0.9683 - recall_4: 0.9621 -
f1_score: 0.9653 - val_loss: 0.6050 - val_categorical_accuracy: 0.8700 -
val_precision_4: 0.8752 - val_recall_4: 0.8633 - val_f1_score: 0.8688
Epoch 27/75
```

```
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1283 -
categorical_accuracy: 0.9607 - precision_4: 0.9645 - recall_4: 0.9567 -
f1_score: 0.9607 - val_loss: 0.5280 - val_categorical_accuracy: 0.8542 -
val_precision_4: 0.8622 - val_recall_4: 0.8460 - val_f1_score: 0.8544
Epoch 28/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1112 -
categorical_accuracy: 0.9659 - precision_4: 0.9685 - recall_4: 0.9626 -
f1_score: 0.9659 - val_loss: 0.6938 - val_categorical_accuracy: 0.8490 -
val_precision_4: 0.8602 - val_recall_4: 0.8421 - val_f1_score: 0.8478
Epoch 29/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1197 -
categorical_accuracy: 0.9644 - precision_4: 0.9680 - recall_4: 0.9614 -
f1_score: 0.9644 - val_loss: 0.6491 - val_categorical_accuracy: 0.8740 -
val_precision_4: 0.8786 - val_recall_4: 0.8673 - val_f1_score: 0.8739
Epoch 30/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1068 -
categorical_accuracy: 0.9695 - precision_4: 0.9719 - recall_4: 0.9671 -
f1_score: 0.9695 - val_loss: 0.5848 - val_categorical_accuracy: 0.8631 -
val_precision_4: 0.8690 - val_recall_4: 0.8567 - val_f1_score: 0.8636
Epoch 31/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1115 -
categorical_accuracy: 0.9674 - precision_4: 0.9704 - recall_4: 0.9647 -
f1_score: 0.9674 - val_loss: 0.5663 - val_categorical_accuracy: 0.8596 -
val_precision_4: 0.8674 - val_recall_4: 0.8531 - val_f1_score: 0.8602
Epoch 32/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0980 -
categorical_accuracy: 0.9706 - precision_4: 0.9736 - recall_4: 0.9681 -
f1_score: 0.9706 - val_loss: 0.7773 - val_categorical_accuracy: 0.8619 -
val_precision_4: 0.8680 - val_recall_4: 0.8548 - val_f1_score: 0.8616
Epoch 33/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0986 -
categorical_accuracy: 0.9709 - precision_4: 0.9732 - recall_4: 0.9689 -
f1_score: 0.9709 - val_loss: 0.5679 - val_categorical_accuracy: 0.8717 -
val_precision_4: 0.8810 - val_recall_4: 0.8656 - val_f1_score: 0.8718
Epoch 34/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1169 -
categorical_accuracy: 0.9666 - precision_4: 0.9698 - recall_4: 0.9638 -
f1_score: 0.9667 - val_loss: 0.6100 - val_categorical_accuracy: 0.8673 -
val_precision_4: 0.8754 - val_recall_4: 0.8623 - val_f1_score: 0.8660
Epoch 35/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0908 -
categorical_accuracy: 0.9739 - precision_4: 0.9761 - recall_4: 0.9718 -
f1_score: 0.9739 - val_loss: 0.6531 - val_categorical_accuracy: 0.8702 -
val_precision_4: 0.8732 - val_recall_4: 0.8654 - val_f1_score: 0.8695
Epoch 36/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0886 -
categorical_accuracy: 0.9746 - precision_4: 0.9763 - recall_4: 0.9730 -
f1_score: 0.9746 - val_loss: 0.7227 - val_categorical_accuracy: 0.8604 -
```

```
val_precision_4: 0.8665 - val_recall_4: 0.8560 - val_f1_score: 0.8612
Epoch 37/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.1110 -
categorical_accuracy: 0.9681 - precision_4: 0.9711 - recall_4: 0.9654 -
f1_score: 0.9682 - val_loss: 0.6514 - val_categorical_accuracy: 0.8587 -
val_precision_4: 0.8657 - val_recall_4: 0.8512 - val_f1_score: 0.8568
Epoch 38/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0952 -
categorical_accuracy: 0.9723 - precision_4: 0.9746 - recall_4: 0.9704 -
f1_score: 0.9723 - val_loss: 0.6073 - val_categorical_accuracy: 0.8729 -
val_precision_4: 0.8788 - val_recall_4: 0.8690 - val_f1_score: 0.8728
Epoch 39/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0913 -
categorical_accuracy: 0.9720 - precision_4: 0.9746 - recall_4: 0.9699 -
f1_score: 0.9721 - val_loss: 0.7151 - val_categorical_accuracy: 0.8715 -
val_precision_4: 0.8770 - val_recall_4: 0.8658 - val_f1_score: 0.8703
Epoch 40/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0880 -
categorical_accuracy: 0.9750 - precision_4: 0.9770 - recall_4: 0.9732 -
f1_score: 0.9750 - val_loss: 0.5798 - val_categorical_accuracy: 0.8783 -
val_precision_4: 0.8840 - val_recall_4: 0.8735 - val_f1_score: 0.8778
Epoch 41/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0788 -
categorical_accuracy: 0.9765 - precision_4: 0.9783 - recall_4: 0.9747 -
f1_score: 0.9765 - val_loss: 0.6586 - val_categorical_accuracy: 0.8710 -
val_precision_4: 0.8749 - val_recall_4: 0.8681 - val_f1_score: 0.8695
Epoch 42/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0806 -
categorical_accuracy: 0.9753 - precision_4: 0.9775 - recall_4: 0.9732 -
f1_score: 0.9754 - val_loss: 0.8459 - val_categorical_accuracy: 0.8438 -
val_precision_4: 0.8479 - val_recall_4: 0.8348 - val_f1_score: 0.8453
Epoch 43/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0817 -
categorical_accuracy: 0.9773 - precision_4: 0.9790 - recall_4: 0.9752 -
f1_score: 0.9773 - val_loss: 0.7015 - val_categorical_accuracy: 0.8681 -
val_precision_4: 0.8727 - val_recall_4: 0.8625 - val_f1_score: 0.8678
Epoch 44/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0930 -
categorical_accuracy: 0.9741 - precision_4: 0.9764 - recall_4: 0.9719 -
f1_score: 0.9741 - val_loss: 0.7624 - val_categorical_accuracy: 0.8715 -
val_precision_4: 0.8759 - val_recall_4: 0.8660 - val_f1_score: 0.8703
Epoch 45/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0830 -
categorical_accuracy: 0.9771 - precision_4: 0.9791 - recall_4: 0.9749 -
f1_score: 0.9771 - val_loss: 0.7200 - val_categorical_accuracy: 0.8721 -
val_precision_4: 0.8751 - val_recall_4: 0.8669 - val_f1_score: 0.8719
Epoch 46/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0890 -
```

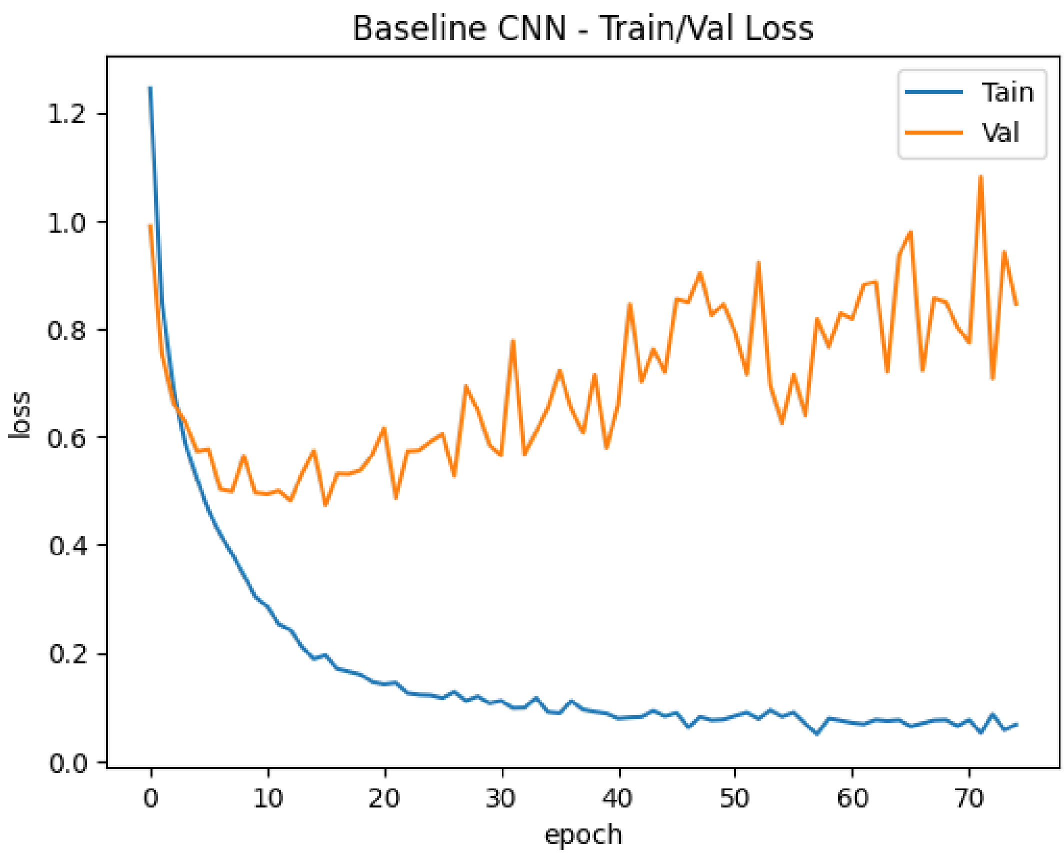
```
categorical_accuracy: 0.9759 - precision_4: 0.9784 - recall_4: 0.9737 -
f1_score: 0.9759 - val_loss: 0.8551 - val_categorical_accuracy: 0.8669 -
val_precision_4: 0.8708 - val_recall_4: 0.8619 - val_f1_score: 0.8648
Epoch 47/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0620 -
categorical_accuracy: 0.9822 - precision_4: 0.9837 - recall_4: 0.9810 -
f1_score: 0.9823 - val_loss: 0.8494 - val_categorical_accuracy: 0.8846 -
val_precision_4: 0.8870 - val_recall_4: 0.8802 - val_f1_score: 0.8843
Epoch 48/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0816 -
categorical_accuracy: 0.9771 - precision_4: 0.9793 - recall_4: 0.9750 -
f1_score: 0.9772 - val_loss: 0.9036 - val_categorical_accuracy: 0.8717 -
val_precision_4: 0.8747 - val_recall_4: 0.8683 - val_f1_score: 0.8719
Epoch 49/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0756 -
categorical_accuracy: 0.9792 - precision_4: 0.9804 - recall_4: 0.9780 -
f1_score: 0.9792 - val_loss: 0.8252 - val_categorical_accuracy: 0.8692 -
val_precision_4: 0.8725 - val_recall_4: 0.8627 - val_f1_score: 0.8687
Epoch 50/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0768 -
categorical_accuracy: 0.9771 - precision_4: 0.9788 - recall_4: 0.9751 -
f1_score: 0.9771 - val_loss: 0.8460 - val_categorical_accuracy: 0.8685 -
val_precision_4: 0.8730 - val_recall_4: 0.8650 - val_f1_score: 0.8684
Epoch 51/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0835 -
categorical_accuracy: 0.9770 - precision_4: 0.9788 - recall_4: 0.9756 -
f1_score: 0.9771 - val_loss: 0.7936 - val_categorical_accuracy: 0.8650 -
val_precision_4: 0.8708 - val_recall_4: 0.8608 - val_f1_score: 0.8651
Epoch 52/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0898 -
categorical_accuracy: 0.9760 - precision_4: 0.9781 - recall_4: 0.9743 -
f1_score: 0.9761 - val_loss: 0.7157 - val_categorical_accuracy: 0.8585 -
val_precision_4: 0.8634 - val_recall_4: 0.8531 - val_f1_score: 0.8595
Epoch 53/75
1200/1200 [==============================] - 9s 7ms/step - loss: 0.0777 -
categorical_accuracy: 0.9796 - precision_4: 0.9813 - recall_4: 0.9779 -
f1_score: 0.9796 - val_loss: 0.9226 - val_categorical_accuracy: 0.8692 -
val_precision_4: 0.8738 - val_recall_4: 0.8656 - val_f1_score: 0.8678
Epoch 54/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0942 -
categorical_accuracy: 0.9772 - precision_4: 0.9791 - recall_4: 0.9747 -
f1_score: 0.9773 - val_loss: 0.6954 - val_categorical_accuracy: 0.8562 -
val_precision_4: 0.8700 - val_recall_4: 0.8465 - val_f1_score: 0.8559
Epoch 55/75
1200/1200 [==============================] - 9s 7ms/step - loss: 0.0820 -
categorical_accuracy: 0.9789 - precision_4: 0.9812 - recall_4: 0.9768 -
f1_score: 0.9789 - val_loss: 0.6249 - val_categorical_accuracy: 0.8685 -
val_precision_4: 0.8767 - val_recall_4: 0.8619 - val_f1_score: 0.8685
```

```
Epoch 56/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0899 -
categorical_accuracy: 0.9742 - precision_4: 0.9769 - recall_4: 0.9715 -
f1_score: 0.9742 - val_loss: 0.7158 - val_categorical_accuracy: 0.8648 -
val_precision_4: 0.8707 - val_recall_4: 0.8583 - val_f1_score: 0.8646
Epoch 57/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0688 -
categorical_accuracy: 0.9815 - precision_4: 0.9829 - recall_4: 0.9797 -
f1_score: 0.9815 - val_loss: 0.6395 - val_categorical_accuracy: 0.8842 -
val_precision_4: 0.8901 - val_recall_4: 0.8788 - val_f1_score: 0.8838
Epoch 58/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0495 -
categorical_accuracy: 0.9872 - precision_4: 0.9881 - recall_4: 0.9862 -
f1_score: 0.9872 - val_loss: 0.8185 - val_categorical_accuracy: 0.8788 -
val_precision_4: 0.8835 - val_recall_4: 0.8754 - val_f1_score: 0.8784
Epoch 59/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0789 -
categorical_accuracy: 0.9780 - precision_4: 0.9798 - recall_4: 0.9762 -
f1_score: 0.9780 - val_loss: 0.7660 - val_categorical_accuracy: 0.8765 -
val_precision_4: 0.8825 - val_recall_4: 0.8729 - val_f1_score: 0.8761
Epoch 60/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0749 -
categorical_accuracy: 0.9794 - precision_4: 0.9816 - recall_4: 0.9772 -
f1_score: 0.9794 - val_loss: 0.8287 - val_categorical_accuracy: 0.8756 -
val_precision_4: 0.8823 - val_recall_4: 0.8712 - val_f1_score: 0.8750
Epoch 61/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0706 -
categorical_accuracy: 0.9811 - precision_4: 0.9825 - recall_4: 0.9795 -
f1_score: 0.9811 - val_loss: 0.8182 - val_categorical_accuracy: 0.8794 -
val_precision_4: 0.8851 - val_recall_4: 0.8748 - val_f1_score: 0.8793
Epoch 62/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0682 -
categorical_accuracy: 0.9829 - precision_4: 0.9842 - recall_4: 0.9816 -
f1_score: 0.9830 - val_loss: 0.8818 - val_categorical_accuracy: 0.8725 -
val_precision_4: 0.8791 - val_recall_4: 0.8679 - val_f1_score: 0.8718
Epoch 63/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0763 -
categorical_accuracy: 0.9789 - precision_4: 0.9809 - recall_4: 0.9773 -
f1_score: 0.9790 - val_loss: 0.8872 - val_categorical_accuracy: 0.8756 -
val_precision_4: 0.8819 - val_recall_4: 0.8700 - val_f1_score: 0.8747
Epoch 64/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0741 -
categorical_accuracy: 0.9807 - precision_4: 0.9825 - recall_4: 0.9792 -
f1_score: 0.9807 - val_loss: 0.7208 - val_categorical_accuracy: 0.8619 -
val_precision_4: 0.8691 - val_recall_4: 0.8537 - val_f1_score: 0.8616
Epoch 65/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0759 -
categorical_accuracy: 0.9798 - precision_4: 0.9813 - recall_4: 0.9786 -
```

```
f1_score: 0.9799 - val_loss: 0.9367 - val_categorical_accuracy: 0.8767 -
val_precision_4: 0.8804 - val_recall_4: 0.8723 - val_f1_score: 0.8766
Epoch 66/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0640 -
categorical_accuracy: 0.9832 - precision_4: 0.9846 - recall_4: 0.9818 -
f1_score: 0.9832 - val_loss: 0.9798 - val_categorical_accuracy: 0.8679 -
val_precision_4: 0.8752 - val_recall_4: 0.8621 - val_f1_score: 0.8660
Epoch 67/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0697 -
categorical_accuracy: 0.9815 - precision_4: 0.9832 - recall_4: 0.9800 -
f1_score: 0.9815 - val_loss: 0.7233 - val_categorical_accuracy: 0.8725 -
val_precision_4: 0.8761 - val_recall_4: 0.8687 - val_f1_score: 0.8739
Epoch 68/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0753 -
categorical_accuracy: 0.9817 - precision_4: 0.9841 - recall_4: 0.9800 -
f1_score: 0.9817 - val_loss: 0.8569 - val_categorical_accuracy: 0.8758 -
val_precision_4: 0.8812 - val_recall_4: 0.8702 - val_f1_score: 0.8765
Epoch 69/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0761 -
categorical_accuracy: 0.9806 - precision_4: 0.9823 - recall_4: 0.9786 -
f1_score: 0.9806 - val_loss: 0.8497 - val_categorical_accuracy: 0.8788 -
val_precision_4: 0.8835 - val_recall_4: 0.8754 - val_f1_score: 0.8777
Epoch 70/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0648 -
categorical_accuracy: 0.9843 - precision_4: 0.9859 - recall_4: 0.9829 -
f1_score: 0.9843 - val_loss: 0.8029 - val_categorical_accuracy: 0.8771 -
val_precision_4: 0.8808 - val_recall_4: 0.8744 - val_f1_score: 0.8767
Epoch 71/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0760 -
categorical_accuracy: 0.9806 - precision_4: 0.9825 - recall_4: 0.9785 -
f1_score: 0.9806 - val_loss: 0.7742 - val_categorical_accuracy: 0.8779 -
val_precision_4: 0.8831 - val_recall_4: 0.8737 - val_f1_score: 0.8780
Epoch 72/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0518 -
categorical_accuracy: 0.9868 - precision_4: 0.9880 - recall_4: 0.9855 -
f1_score: 0.9869 - val_loss: 1.0814 - val_categorical_accuracy: 0.8679 -
val_precision_4: 0.8764 - val_recall_4: 0.8625 - val_f1_score: 0.8662
Epoch 73/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0864 -
categorical_accuracy: 0.9782 - precision_4: 0.9806 - recall_4: 0.9761 -
f1_score: 0.9783 - val_loss: 0.7088 - val_categorical_accuracy: 0.8802 -
val_precision_4: 0.8844 - val_recall_4: 0.8769 - val_f1_score: 0.8805
Epoch 74/75
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0575 -
categorical_accuracy: 0.9863 - precision_4: 0.9876 - recall_4: 0.9852 -
f1_score: 0.9863 - val_loss: 0.9422 - val_categorical_accuracy: 0.8760 -
val_precision_4: 0.8822 - val_recall_4: 0.8719 - val_f1_score: 0.8749
Epoch 75/75
```

```
1200/1200 [==============================] - 8s 7ms/step - loss: 0.0674 -
categorical_accuracy: 0.9827 - precision_4: 0.9845 - recall_4: 0.9811 -
f1_score: 0.9828 - val_loss: 0.8467 - val_categorical_accuracy: 0.8815 -
val_precision_4: 0.8850 - val_recall_4: 0.8754 - val_f1_score: 0.8810
```

[ ]:
```
#plot loss
plt.plot(history_cnn_baseline.history['loss'])
plt.plot(history_cnn_baseline.history['val_loss'])
plt.title('Baseline CNN - Train/Val Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Tain', 'Val'])
plt.show()
```



Improved CNN Model 1 This CNN will add additional convolutional layers (5) and start with a higher number of units per layer (512). The units will gradually decrease with each convolutional layer down to 32. Dropout is also added to both the convolutional layers and the fully connected layers.

```
[ ]: # free up resources
gc.collect()

# setup checkpoint
checkpoint_filepath_cnn2 = '/content/drive/MyDrive/USD/models/
 ↪composer-classifier/cnn-2'
model_checkpoint_callback_cnn2 = keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath_cnn2,
    monitor='val_categorical_accuracy',
    mode='max',
    save_best_only=True)

cnn_exp1 = tf.keras.Sequential([

  tf.keras.layers.Normalization(axis=None),

  tf.keras.layers.Conv1D(512, kernel_size=3, activation='relu',␣
 ↪padding='causal'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Conv1D(256, kernel_size=3, activation='relu',␣
 ↪padding='causal'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Conv1D(128, kernel_size=3, activation='relu',␣
 ↪padding='causal'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Conv1D(64, kernel_size=3, activation='relu',␣
 ↪padding='causal'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Conv1D(32, kernel_size=3, activation='relu',␣
 ↪padding='causal'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.MaxPooling1D(2),

  tf.keras.layers.Flatten(),

  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(64, activation='relu'),
  tf.keras.layers.Dropout(0.2),
```

```python
  tf.keras.layers.Dense(32, activation='relu'),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(NUM_COMPOSERS, activation='softmax')
])

# Compile the model
cnn_exp1.compile(
    optimizer=keras.optimizers.Adam(learning_rate=LEARNING_RATE),
    loss=tf.keras.losses.CategoricalCrossentropy(),
    metrics=[keras.metrics.CategoricalAccuracy(), keras.metrics.Precision(),␣
 ↪keras.metrics.Recall(), keras.metrics.F1Score()]
)
```

[ ]:
```python
# Train the model
history_cnn_exp1 = cnn_exp1.fit(X_train, y_train,␣
 ↪validation_data=(X_val,y_val), epochs=NUM_EPOCHS, batch_size=BATCH_SIZE,␣
 ↪callbacks=[model_checkpoint_callback_cnn2])
```

```
Epoch 1/75
1350/1350 [==============================] - 22s 13ms/step - loss: 1.3635 -
categorical_accuracy: 0.3439 - precision_3: 0.5456 - recall_3: 0.0542 -
f1_score: 0.3053 - val_loss: 1.1748 - val_categorical_accuracy: 0.4778 -
val_precision_3: 0.8850 - val_recall_3: 0.0513 - val_f1_score: 0.4288
Epoch 2/75
1350/1350 [==============================] - 16s 12ms/step - loss: 1.0466 -
categorical_accuracy: 0.5218 - precision_3: 0.6950 - recall_3: 0.2903 -
f1_score: 0.4881 - val_loss: 0.9241 - val_categorical_accuracy: 0.6135 -
val_precision_3: 0.7625 - val_recall_3: 0.3467 - val_f1_score: 0.5744
Epoch 3/75
1350/1350 [==============================] - 17s 12ms/step - loss: 0.8556 -
categorical_accuracy: 0.6391 - precision_3: 0.7458 - recall_3: 0.4637 -
f1_score: 0.6220 - val_loss: 0.8879 - val_categorical_accuracy: 0.6269 -
val_precision_3: 0.7274 - val_recall_3: 0.4472 - val_f1_score: 0.5938
Epoch 4/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.7278 -
categorical_accuracy: 0.6997 - precision_3: 0.7674 - recall_3: 0.5873 -
f1_score: 0.6893 - val_loss: 0.6964 - val_categorical_accuracy: 0.7335 -
val_precision_3: 0.8058 - val_recall_3: 0.6094 - val_f1_score: 0.7246
Epoch 5/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.6528 -
categorical_accuracy: 0.7361 - precision_3: 0.7844 - recall_3: 0.6619 -
f1_score: 0.7282 - val_loss: 0.6768 - val_categorical_accuracy: 0.7335 -
val_precision_3: 0.7779 - val_recall_3: 0.6570 - val_f1_score: 0.7127
Epoch 6/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.6043 -
categorical_accuracy: 0.7580 - precision_3: 0.7977 - recall_3: 0.6997 -
f1_score: 0.7517 - val_loss: 0.6863 - val_categorical_accuracy: 0.7411 -
val_precision_3: 0.7871 - val_recall_3: 0.6750 - val_f1_score: 0.7305
```

```
Epoch 7/75
1350/1350 [==============================] - 17s 12ms/step - loss: 0.5614 -
categorical_accuracy: 0.7768 - precision_3: 0.8079 - recall_3: 0.7321 -
f1_score: 0.7712 - val_loss: 0.6206 - val_categorical_accuracy: 0.7711 -
val_precision_3: 0.7969 - val_recall_3: 0.7231 - val_f1_score: 0.7634
Epoch 8/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.5260 -
categorical_accuracy: 0.7949 - precision_3: 0.8244 - recall_3: 0.7563 -
f1_score: 0.7898 - val_loss: 0.6067 - val_categorical_accuracy: 0.7787 -
val_precision_3: 0.8084 - val_recall_3: 0.7211 - val_f1_score: 0.7681
Epoch 9/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.4894 -
categorical_accuracy: 0.8084 - precision_3: 0.8346 - recall_3: 0.7765 -
f1_score: 0.8037 - val_loss: 0.5098 - val_categorical_accuracy: 0.8109 -
val_precision_3: 0.8441 - val_recall_3: 0.7711 - val_f1_score: 0.8030
Epoch 10/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.4745 -
categorical_accuracy: 0.8187 - precision_3: 0.8454 - recall_3: 0.7885 -
f1_score: 0.8145 - val_loss: 0.5357 - val_categorical_accuracy: 0.8063 -
val_precision_3: 0.8294 - val_recall_3: 0.7698 - val_f1_score: 0.7968
Epoch 11/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.4432 -
categorical_accuracy: 0.8300 - precision_3: 0.8547 - recall_3: 0.8025 -
f1_score: 0.8258 - val_loss: 0.5106 - val_categorical_accuracy: 0.8178 -
val_precision_3: 0.8500 - val_recall_3: 0.7661 - val_f1_score: 0.8079
Epoch 12/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.4245 -
categorical_accuracy: 0.8395 - precision_3: 0.8631 - recall_3: 0.8139 -
f1_score: 0.8361 - val_loss: 0.5524 - val_categorical_accuracy: 0.7933 -
val_precision_3: 0.8213 - val_recall_3: 0.7543 - val_f1_score: 0.7774
Epoch 13/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.4051 -
categorical_accuracy: 0.8479 - precision_3: 0.8697 - recall_3: 0.8236 -
f1_score: 0.8445 - val_loss: 0.4690 - val_categorical_accuracy: 0.8393 -
val_precision_3: 0.8718 - val_recall_3: 0.7919 - val_f1_score: 0.8339
Epoch 14/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.3940 -
categorical_accuracy: 0.8561 - precision_3: 0.8780 - recall_3: 0.8332 -
f1_score: 0.8531 - val_loss: 0.5536 - val_categorical_accuracy: 0.8083 -
val_precision_3: 0.8359 - val_recall_3: 0.7709 - val_f1_score: 0.8015
Epoch 15/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.3761 -
categorical_accuracy: 0.8609 - precision_3: 0.8827 - recall_3: 0.8386 -
f1_score: 0.8581 - val_loss: 0.5237 - val_categorical_accuracy: 0.8111 -
val_precision_3: 0.8360 - val_recall_3: 0.7789 - val_f1_score: 0.8051
Epoch 16/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.3529 -
categorical_accuracy: 0.8726 - precision_3: 0.8922 - recall_3: 0.8534 -
```

```
f1_score: 0.8701 - val_loss: 0.5200 - val_categorical_accuracy: 0.8272 -
val_precision_3: 0.8564 - val_recall_3: 0.7678 - val_f1_score: 0.8201
Epoch 17/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.3377 -
categorical_accuracy: 0.8799 - precision_3: 0.8976 - recall_3: 0.8609 -
f1_score: 0.8774 - val_loss: 0.4649 - val_categorical_accuracy: 0.8356 -
val_precision_3: 0.8663 - val_recall_3: 0.7993 - val_f1_score: 0.8276
Epoch 18/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.3371 -
categorical_accuracy: 0.8799 - precision_3: 0.8983 - recall_3: 0.8603 -
f1_score: 0.8776 - val_loss: 0.4663 - val_categorical_accuracy: 0.8394 -
val_precision_3: 0.8725 - val_recall_3: 0.8022 - val_f1_score: 0.8327
Epoch 19/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.3178 -
categorical_accuracy: 0.8865 - precision_3: 0.9016 - recall_3: 0.8697 -
f1_score: 0.8843 - val_loss: 0.4495 - val_categorical_accuracy: 0.8494 -
val_precision_3: 0.8703 - val_recall_3: 0.8317 - val_f1_score: 0.8415
Epoch 20/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.3080 -
categorical_accuracy: 0.8945 - precision_3: 0.9094 - recall_3: 0.8785 -
f1_score: 0.8924 - val_loss: 0.4542 - val_categorical_accuracy: 0.8622 -
val_precision_3: 0.8898 - val_recall_3: 0.8239 - val_f1_score: 0.8585
Epoch 21/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.3152 -
categorical_accuracy: 0.8900 - precision_3: 0.9057 - recall_3: 0.8739 -
f1_score: 0.8879 - val_loss: 0.4255 - val_categorical_accuracy: 0.8578 -
val_precision_3: 0.8849 - val_recall_3: 0.8302 - val_f1_score: 0.8522
Epoch 22/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2751 -
categorical_accuracy: 0.9031 - precision_3: 0.9164 - recall_3: 0.8904 -
f1_score: 0.9010 - val_loss: 0.4330 - val_categorical_accuracy: 0.8535 -
val_precision_3: 0.8773 - val_recall_3: 0.8326 - val_f1_score: 0.8467
Epoch 23/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.2808 -
categorical_accuracy: 0.9026 - precision_3: 0.9158 - recall_3: 0.8885 -
f1_score: 0.9006 - val_loss: 0.4103 - val_categorical_accuracy: 0.8726 -
val_precision_3: 0.8935 - val_recall_3: 0.8469 - val_f1_score: 0.8669
Epoch 24/75
1350/1350 [==============================] - 17s 12ms/step - loss: 0.2892 -
categorical_accuracy: 0.9036 - precision_3: 0.9175 - recall_3: 0.8891 -
f1_score: 0.9017 - val_loss: 0.3689 - val_categorical_accuracy: 0.8783 -
val_precision_3: 0.8964 - val_recall_3: 0.8576 - val_f1_score: 0.8741
Epoch 25/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2664 -
categorical_accuracy: 0.9105 - precision_3: 0.9243 - recall_3: 0.8974 -
f1_score: 0.9084 - val_loss: 0.4466 - val_categorical_accuracy: 0.8591 -
val_precision_3: 0.8812 - val_recall_3: 0.8243 - val_f1_score: 0.8549
Epoch 26/75
```

```
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2543 -
categorical_accuracy: 0.9124 - precision_3: 0.9260 - recall_3: 0.8992 -
f1_score: 0.9105 - val_loss: 0.4598 - val_categorical_accuracy: 0.8565 -
val_precision_3: 0.8758 - val_recall_3: 0.8307 - val_f1_score: 0.8501
Epoch 27/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2610 -
categorical_accuracy: 0.9147 - precision_3: 0.9272 - recall_3: 0.9016 -
f1_score: 0.9129 - val_loss: 0.4434 - val_categorical_accuracy: 0.8593 -
val_precision_3: 0.8815 - val_recall_3: 0.8309 - val_f1_score: 0.8529
Epoch 28/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2746 -
categorical_accuracy: 0.9109 - precision_3: 0.9266 - recall_3: 0.8955 -
f1_score: 0.9090 - val_loss: 0.4377 - val_categorical_accuracy: 0.8685 -
val_precision_3: 0.8910 - val_recall_3: 0.8324 - val_f1_score: 0.8644
Epoch 29/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.2292 -
categorical_accuracy: 0.9228 - precision_3: 0.9346 - recall_3: 0.9107 -
f1_score: 0.9213 - val_loss: 0.3722 - val_categorical_accuracy: 0.8900 -
val_precision_3: 0.9096 - val_recall_3: 0.8663 - val_f1_score: 0.8860
Epoch 30/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2287 -
categorical_accuracy: 0.9246 - precision_3: 0.9355 - recall_3: 0.9132 -
f1_score: 0.9229 - val_loss: 0.3854 - val_categorical_accuracy: 0.8859 -
val_precision_3: 0.9013 - val_recall_3: 0.8691 - val_f1_score: 0.8823
Epoch 31/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.2413 -
categorical_accuracy: 0.9210 - precision_3: 0.9323 - recall_3: 0.9095 -
f1_score: 0.9195 - val_loss: 0.3638 - val_categorical_accuracy: 0.8917 -
val_precision_3: 0.9125 - val_recall_3: 0.8654 - val_f1_score: 0.8882
Epoch 32/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2429 -
categorical_accuracy: 0.9216 - precision_3: 0.9344 - recall_3: 0.9078 -
f1_score: 0.9201 - val_loss: 0.3875 - val_categorical_accuracy: 0.8787 -
val_precision_3: 0.8982 - val_recall_3: 0.8561 - val_f1_score: 0.8740
Epoch 33/75
1350/1350 [==============================] - 17s 12ms/step - loss: 0.2248 -
categorical_accuracy: 0.9275 - precision_3: 0.9388 - recall_3: 0.9152 -
f1_score: 0.9260 - val_loss: 0.3647 - val_categorical_accuracy: 0.8920 -
val_precision_3: 0.9073 - val_recall_3: 0.8750 - val_f1_score: 0.8884
Epoch 34/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2179 -
categorical_accuracy: 0.9309 - precision_3: 0.9420 - recall_3: 0.9202 -
f1_score: 0.9294 - val_loss: 0.4086 - val_categorical_accuracy: 0.8696 -
val_precision_3: 0.8943 - val_recall_3: 0.8363 - val_f1_score: 0.8640
Epoch 35/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2261 -
categorical_accuracy: 0.9273 - precision_3: 0.9397 - recall_3: 0.9155 -
f1_score: 0.9259 - val_loss: 0.4531 - val_categorical_accuracy: 0.8543 -
```

```
val_precision_3: 0.8753 - val_recall_3: 0.8291 - val_f1_score: 0.8484
Epoch 36/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.2140 -
categorical_accuracy: 0.9315 - precision_3: 0.9421 - recall_3: 0.9212 -
f1_score: 0.9301 - val_loss: 0.3522 - val_categorical_accuracy: 0.8930 -
val_precision_3: 0.9147 - val_recall_3: 0.8696 - val_f1_score: 0.8893
Epoch 37/75
1350/1350 [==============================] - 17s 12ms/step - loss: 0.1919 -
categorical_accuracy: 0.9398 - precision_3: 0.9493 - recall_3: 0.9300 -
f1_score: 0.9387 - val_loss: 0.3570 - val_categorical_accuracy: 0.9006 -
val_precision_3: 0.9227 - val_recall_3: 0.8757 - val_f1_score: 0.8972
Epoch 38/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2395 -
categorical_accuracy: 0.9228 - precision_3: 0.9358 - recall_3: 0.9092 -
f1_score: 0.9216 - val_loss: 0.4026 - val_categorical_accuracy: 0.8756 -
val_precision_3: 0.8928 - val_recall_3: 0.8602 - val_f1_score: 0.8710
Epoch 39/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2063 -
categorical_accuracy: 0.9343 - precision_3: 0.9460 - recall_3: 0.9220 -
f1_score: 0.9330 - val_loss: 0.3711 - val_categorical_accuracy: 0.8893 -
val_precision_3: 0.9030 - val_recall_3: 0.8772 - val_f1_score: 0.8851
Epoch 40/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2078 -
categorical_accuracy: 0.9328 - precision_3: 0.9445 - recall_3: 0.9208 -
f1_score: 0.9315 - val_loss: 0.4238 - val_categorical_accuracy: 0.8789 -
val_precision_3: 0.8967 - val_recall_3: 0.8572 - val_f1_score: 0.8746
Epoch 41/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2112 -
categorical_accuracy: 0.9353 - precision_3: 0.9455 - recall_3: 0.9236 -
f1_score: 0.9341 - val_loss: 0.3791 - val_categorical_accuracy: 0.8894 -
val_precision_3: 0.9036 - val_recall_3: 0.8746 - val_f1_score: 0.8858
Epoch 42/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1826 -
categorical_accuracy: 0.9434 - precision_3: 0.9530 - recall_3: 0.9336 -
f1_score: 0.9423 - val_loss: 0.4229 - val_categorical_accuracy: 0.8693 -
val_precision_3: 0.9009 - val_recall_3: 0.8313 - val_f1_score: 0.8646
Epoch 43/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1893 -
categorical_accuracy: 0.9436 - precision_3: 0.9535 - recall_3: 0.9316 -
f1_score: 0.9424 - val_loss: 0.3777 - val_categorical_accuracy: 0.8850 -
val_precision_3: 0.9207 - val_recall_3: 0.8380 - val_f1_score: 0.8817
Epoch 44/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2266 -
categorical_accuracy: 0.9291 - precision_3: 0.9418 - recall_3: 0.9166 -
f1_score: 0.9277 - val_loss: 0.4153 - val_categorical_accuracy: 0.8802 -
val_precision_3: 0.8993 - val_recall_3: 0.8533 - val_f1_score: 0.8767
Epoch 45/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2057 -
```

```
categorical_accuracy: 0.9383 - precision_3: 0.9490 - recall_3: 0.9273 -
f1_score: 0.9374 - val_loss: 0.3711 - val_categorical_accuracy: 0.8848 -
val_precision_3: 0.9060 - val_recall_3: 0.8652 - val_f1_score: 0.8814
Epoch 46/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1913 -
categorical_accuracy: 0.9418 - precision_3: 0.9515 - recall_3: 0.9321 -
f1_score: 0.9409 - val_loss: 0.3780 - val_categorical_accuracy: 0.8926 -
val_precision_3: 0.9108 - val_recall_3: 0.8757 - val_f1_score: 0.8896
Epoch 47/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1877 -
categorical_accuracy: 0.9429 - precision_3: 0.9528 - recall_3: 0.9331 -
f1_score: 0.9420 - val_loss: 0.4092 - val_categorical_accuracy: 0.8876 -
val_precision_3: 0.9199 - val_recall_3: 0.8528 - val_f1_score: 0.8839
Epoch 48/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1827 -
categorical_accuracy: 0.9440 - precision_3: 0.9535 - recall_3: 0.9339 -
f1_score: 0.9431 - val_loss: 0.3507 - val_categorical_accuracy: 0.8980 -
val_precision_3: 0.9144 - val_recall_3: 0.8820 - val_f1_score: 0.8951
Epoch 49/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2219 -
categorical_accuracy: 0.9359 - precision_3: 0.9479 - recall_3: 0.9216 -
f1_score: 0.9350 - val_loss: 0.3718 - val_categorical_accuracy: 0.8930 -
val_precision_3: 0.9155 - val_recall_3: 0.8704 - val_f1_score: 0.8893
Epoch 50/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.1880 -
categorical_accuracy: 0.9422 - precision_3: 0.9529 - recall_3: 0.9316 -
f1_score: 0.9413 - val_loss: 0.3345 - val_categorical_accuracy: 0.9017 -
val_precision_3: 0.9208 - val_recall_3: 0.8737 - val_f1_score: 0.8984
Epoch 51/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1791 -
categorical_accuracy: 0.9478 - precision_3: 0.9569 - recall_3: 0.9385 -
f1_score: 0.9471 - val_loss: 0.3558 - val_categorical_accuracy: 0.8976 -
val_precision_3: 0.9165 - val_recall_3: 0.8776 - val_f1_score: 0.8941
Epoch 52/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2130 -
categorical_accuracy: 0.9383 - precision_3: 0.9493 - recall_3: 0.9264 -
f1_score: 0.9376 - val_loss: 0.4273 - val_categorical_accuracy: 0.8863 -
val_precision_3: 0.9077 - val_recall_3: 0.8598 - val_f1_score: 0.8815
Epoch 53/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1729 -
categorical_accuracy: 0.9520 - precision_3: 0.9602 - recall_3: 0.9431 -
f1_score: 0.9514 - val_loss: 0.4014 - val_categorical_accuracy: 0.8911 -
val_precision_3: 0.9061 - val_recall_3: 0.8687 - val_f1_score: 0.8863
Epoch 54/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1704 -
categorical_accuracy: 0.9504 - precision_3: 0.9586 - recall_3: 0.9413 -
f1_score: 0.9496 - val_loss: 0.4057 - val_categorical_accuracy: 0.8907 -
val_precision_3: 0.9098 - val_recall_3: 0.8644 - val_f1_score: 0.8874
```

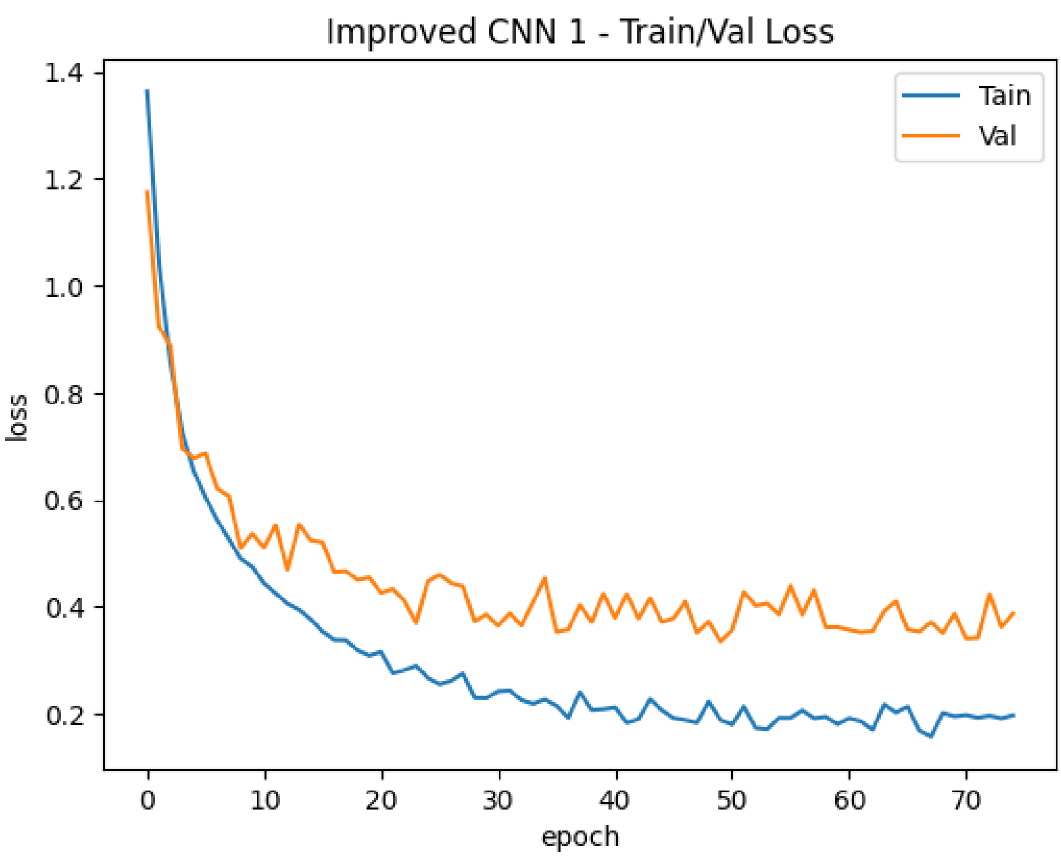
```
Epoch 55/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.1916 -
categorical_accuracy: 0.9433 - precision_3: 0.9533 - recall_3: 0.9327 -
f1_score: 0.9422 - val_loss: 0.3850 - val_categorical_accuracy: 0.9019 -
val_precision_3: 0.9175 - val_recall_3: 0.8752 - val_f1_score: 0.8989
Epoch 56/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1916 -
categorical_accuracy: 0.9420 - precision_3: 0.9528 - recall_3: 0.9315 -
f1_score: 0.9409 - val_loss: 0.4392 - val_categorical_accuracy: 0.8920 -
val_precision_3: 0.9037 - val_recall_3: 0.8737 - val_f1_score: 0.8893
Epoch 57/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.2057 -
categorical_accuracy: 0.9408 - precision_3: 0.9524 - recall_3: 0.9294 -
f1_score: 0.9401 - val_loss: 0.3844 - val_categorical_accuracy: 0.9030 -
val_precision_3: 0.9169 - val_recall_3: 0.8850 - val_f1_score: 0.9007
Epoch 58/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1908 -
categorical_accuracy: 0.9441 - precision_3: 0.9552 - recall_3: 0.9329 -
f1_score: 0.9431 - val_loss: 0.4310 - val_categorical_accuracy: 0.8909 -
val_precision_3: 0.9136 - val_recall_3: 0.8598 - val_f1_score: 0.8872
Epoch 59/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1938 -
categorical_accuracy: 0.9451 - precision_3: 0.9561 - recall_3: 0.9327 -
f1_score: 0.9446 - val_loss: 0.3610 - val_categorical_accuracy: 0.8957 -
val_precision_3: 0.9202 - val_recall_3: 0.8631 - val_f1_score: 0.8923
Epoch 60/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1799 -
categorical_accuracy: 0.9488 - precision_3: 0.9590 - recall_3: 0.9376 -
f1_score: 0.9480 - val_loss: 0.3615 - val_categorical_accuracy: 0.8963 -
val_precision_3: 0.9233 - val_recall_3: 0.8648 - val_f1_score: 0.8927
Epoch 61/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1910 -
categorical_accuracy: 0.9458 - precision_3: 0.9555 - recall_3: 0.9340 -
f1_score: 0.9450 - val_loss: 0.3562 - val_categorical_accuracy: 0.8969 -
val_precision_3: 0.9159 - val_recall_3: 0.8754 - val_f1_score: 0.8921
Epoch 62/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1853 -
categorical_accuracy: 0.9472 - precision_3: 0.9565 - recall_3: 0.9370 -
f1_score: 0.9465 - val_loss: 0.3517 - val_categorical_accuracy: 0.8983 -
val_precision_3: 0.9166 - val_recall_3: 0.8767 - val_f1_score: 0.8945
Epoch 63/75
1350/1350 [==============================] - 17s 12ms/step - loss: 0.1694 -
categorical_accuracy: 0.9514 - precision_3: 0.9608 - recall_3: 0.9419 -
f1_score: 0.9505 - val_loss: 0.3540 - val_categorical_accuracy: 0.9078 -
val_precision_3: 0.9257 - val_recall_3: 0.8881 - val_f1_score: 0.9047
Epoch 64/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2166 -
categorical_accuracy: 0.9407 - precision_3: 0.9517 - recall_3: 0.9280 -
```

```
f1_score: 0.9403 - val_loss: 0.3918 - val_categorical_accuracy: 0.8952 -
val_precision_3: 0.9110 - val_recall_3: 0.8724 - val_f1_score: 0.8925
Epoch 65/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2016 -
categorical_accuracy: 0.9433 - precision_3: 0.9533 - recall_3: 0.9307 -
f1_score: 0.9427 - val_loss: 0.4095 - val_categorical_accuracy: 0.8843 -
val_precision_3: 0.9027 - val_recall_3: 0.8626 - val_f1_score: 0.8810
Epoch 66/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.2126 -
categorical_accuracy: 0.9407 - precision_3: 0.9521 - recall_3: 0.9280 -
f1_score: 0.9400 - val_loss: 0.3570 - val_categorical_accuracy: 0.8950 -
val_precision_3: 0.9166 - val_recall_3: 0.8733 - val_f1_score: 0.8910
Epoch 67/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1676 -
categorical_accuracy: 0.9522 - precision_3: 0.9621 - recall_3: 0.9411 -
f1_score: 0.9516 - val_loss: 0.3527 - val_categorical_accuracy: 0.8981 -
val_precision_3: 0.9179 - val_recall_3: 0.8781 - val_f1_score: 0.8951
Epoch 68/75
1350/1350 [==============================] - 16s 12ms/step - loss: 0.1568 -
categorical_accuracy: 0.9551 - precision_3: 0.9633 - recall_3: 0.9474 -
f1_score: 0.9544 - val_loss: 0.3707 - val_categorical_accuracy: 0.9113 -
val_precision_3: 0.9216 - val_recall_3: 0.8967 - val_f1_score: 0.9078
Epoch 69/75
1350/1350 [==============================] - 14s 11ms/step - loss: 0.2007 -
categorical_accuracy: 0.9426 - precision_3: 0.9550 - recall_3: 0.9300 -
f1_score: 0.9422 - val_loss: 0.3502 - val_categorical_accuracy: 0.9028 -
val_precision_3: 0.9226 - val_recall_3: 0.8828 - val_f1_score: 0.9000
Epoch 70/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1947 -
categorical_accuracy: 0.9459 - precision_3: 0.9550 - recall_3: 0.9354 -
f1_score: 0.9456 - val_loss: 0.3863 - val_categorical_accuracy: 0.9046 -
val_precision_3: 0.9268 - val_recall_3: 0.8819 - val_f1_score: 0.9013
Epoch 71/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1969 -
categorical_accuracy: 0.9449 - precision_3: 0.9549 - recall_3: 0.9348 -
f1_score: 0.9444 - val_loss: 0.3403 - val_categorical_accuracy: 0.9085 -
val_precision_3: 0.9197 - val_recall_3: 0.8926 - val_f1_score: 0.9056
Epoch 72/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1920 -
categorical_accuracy: 0.9466 - precision_3: 0.9570 - recall_3: 0.9360 -
f1_score: 0.9464 - val_loss: 0.3415 - val_categorical_accuracy: 0.9076 -
val_precision_3: 0.9232 - val_recall_3: 0.8928 - val_f1_score: 0.9049
Epoch 73/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1957 -
categorical_accuracy: 0.9453 - precision_3: 0.9563 - recall_3: 0.9333 -
f1_score: 0.9447 - val_loss: 0.4231 - val_categorical_accuracy: 0.8820 -
val_precision_3: 0.8986 - val_recall_3: 0.8554 - val_f1_score: 0.8782
Epoch 74/75
```

```
1350/1350 [==============================] - 14s 11ms/step - loss: 0.1908 -
categorical_accuracy: 0.9442 - precision_3: 0.9550 - recall_3: 0.9344 -
f1_score: 0.9434 - val_loss: 0.3617 - val_categorical_accuracy: 0.9000 -
val_precision_3: 0.9121 - val_recall_3: 0.8854 - val_f1_score: 0.8965
Epoch 75/75
1350/1350 [==============================] - 14s 10ms/step - loss: 0.1965 -
categorical_accuracy: 0.9428 - precision_3: 0.9539 - recall_3: 0.9309 -
f1_score: 0.9422 - val_loss: 0.3869 - val_categorical_accuracy: 0.9028 -
val_precision_3: 0.9242 - val_recall_3: 0.8785 - val_f1_score: 0.9002
```

[ ]:

```python
#plot loss
plt.plot(history_cnn_exp1.history['loss'])
plt.plot(history_cnn_exp1.history['val_loss'])
plt.title('Improved CNN 1 - Train/Val Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Tain', 'Val'])
plt.show()
```



[ ]:
```
# free up resources
gc.collect()
```

[ ]: 63681

[ ]:
```
# load best model from checkpoint
cnn_exp1_best = tf.keras.models.load_model('/content/drive/MyDrive/USD/models/
↪composer-classifier/cnn-2')
```

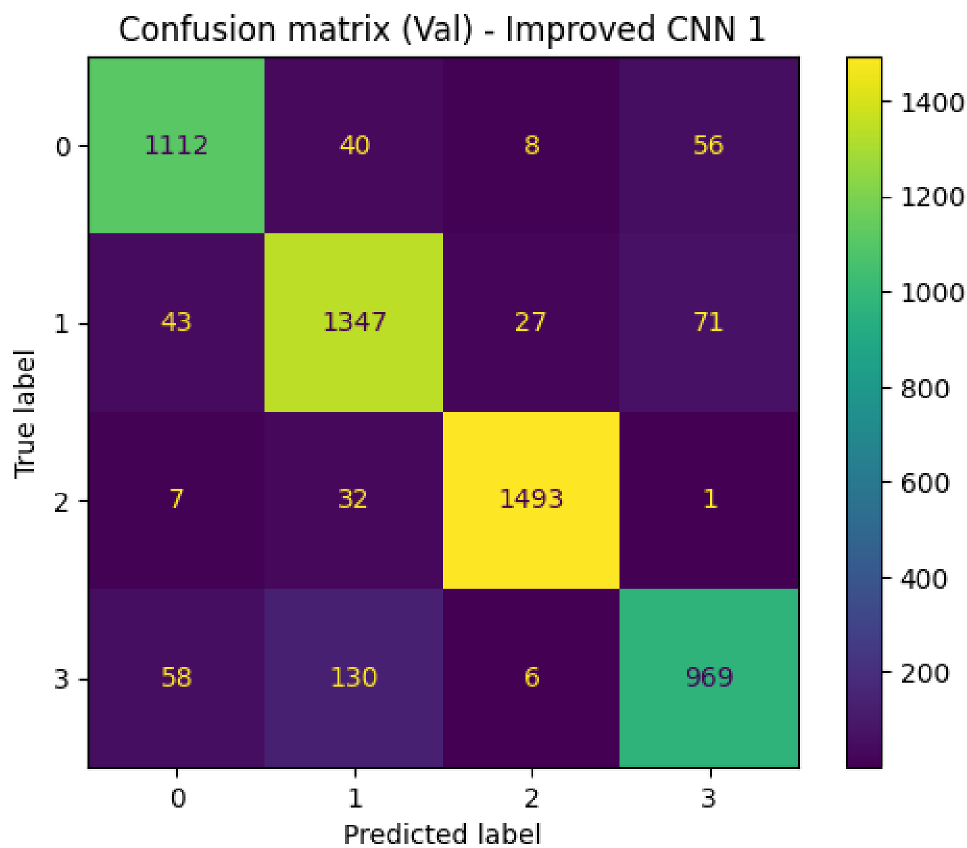
[ ]:
```
# evaluate on val data
loss, accuracy, precision, recall, f1 = cnn_exp1_best.evaluate(X_val, y_val)
print(f'Val Loss: {loss}\nAccuracy: {accuracy}\nPrecision: {precision}\nRecall:␣
↪{recall},\nF1: {f1}')
```

```
169/169 [==============================] - 2s 6ms/step - loss: 0.3707 -
categorical_accuracy: 0.9113 - precision_3: 0.9216 - recall_3: 0.8967 -
f1_score: 0.9078
Val Loss: 0.3707001805305481
Accuracy: 0.9112963080406189
Precision: 0.9215835332870483
Recall: 0.8966666460037231,
F1: [0.91297203 0.8870596  0.97358984 0.85752213]
```

[ ]:
```
# plot confusion matrix
y_pred_cnn1 = cnn_exp1_best.predict(X_val)
cm = confusion_matrix(np.argmax(y_val, axis=1), np.argmax(y_pred_cnn1, axis=1))
ConfusionMatrixDisplay(confusion_matrix=cm).plot();
plt.title('Confusion matrix (Val) - Improved CNN 1')
```

```
169/169 [==============================] - 1s 4ms/step
```

[ ]: Text(0.5, 1.0, 'Confusion matrix (Val) - Improved CNN 1')



Improved CNN Model 2 This CNN will add additional convolutional layers (6) but with a lower number of units per layer to start than Improved CNN-1. The units will also more gradually decrease with each convolutional layer down to 64. Dropout is also increased in both the convolutional layers and the fully connected layers.

[ ]:
```
# free up resources
gc.collect()

# setup checkpoint
checkpoint_filepath = '/content/drive/MyDrive/USD/models/composer-classifier/
 ↪cnn-3'
model_checkpoint_callback = keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    monitor='val_categorical_accuracy',
    mode='max',
    save_best_only=True)

cnn_exp2 = tf.keras.Sequential([
```

```
    tf.keras.layers.Normalization(axis=None),

    tf.keras.layers.Conv1D(256, kernel_size=3, activation='relu', padding='causal'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.MaxPooling1D(2),

    tf.keras.layers.Conv1D(256, kernel_size=3, activation='relu', padding='causal'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.MaxPooling1D(2),

    tf.keras.layers.Conv1D(128, kernel_size=3, activation='relu', padding='causal'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.MaxPooling1D(2),

    tf.keras.layers.Conv1D(128, kernel_size=3, activation='relu', padding='causal'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.MaxPooling1D(2),

    tf.keras.layers.Conv1D(64, kernel_size=3, activation='relu', padding='causal'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.MaxPooling1D(2),

    tf.keras.layers.Conv1D(64, kernel_size=3, activation='relu', padding='causal'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.MaxPooling1D(2),


    tf.keras.layers.Flatten(),

    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.Dense(NUM_COMPOSERS, activation='softmax')
])

# Compile the model
cnn_exp2.compile(
    optimizer=keras.optimizers.Adam(learning_rate=LEARNING_RATE),
    loss=tf.keras.losses.CategoricalCrossentropy(),
```

```
    metrics=[keras.metrics.CategoricalAccuracy(), keras.metrics.Precision(),␣
 ↪keras.metrics.Recall(), keras.metrics.F1Score()]
)
```

[ ]:
```
# Train the model
history_cnn_exp2 = cnn_exp2.fit(X_train, y_train,␣
 ↪validation_data=(X_val,y_val), epochs=NUM_EPOCHS, batch_size=BATCH_SIZE)
```

```
Epoch 1/75
1200/1200 [==============================] - 20s 10ms/step - loss: 1.4117 -
categorical_accuracy: 0.3379 - precision: 0.4945 - recall: 0.0583 - f1_score:
0.3228 - val_loss: 1.2395 - val_categorical_accuracy: 0.3881 - val_precision:
0.8905 - val_recall: 0.0254 - val_f1_score: 0.3389
Epoch 2/75
1200/1200 [==============================] - 10s 8ms/step - loss: 1.0812 -
categorical_accuracy: 0.5107 - precision: 0.6998 - recall: 0.2560 - f1_score:
0.4903 - val_loss: 1.1539 - val_categorical_accuracy: 0.4256 - val_precision:
0.6765 - val_recall: 0.2296 - val_f1_score: 0.3737
Epoch 3/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.9329 -
categorical_accuracy: 0.5986 - precision: 0.7199 - recall: 0.3997 - f1_score:
0.5879 - val_loss: 1.1721 - val_categorical_accuracy: 0.4798 - val_precision:
0.6828 - val_recall: 0.2560 - val_f1_score: 0.4396
Epoch 4/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.8414 -
categorical_accuracy: 0.6543 - precision: 0.7436 - recall: 0.5025 - f1_score:
0.6461 - val_loss: 1.0956 - val_categorical_accuracy: 0.5200 - val_precision:
0.7062 - val_recall: 0.3371 - val_f1_score: 0.4824
Epoch 5/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.7721 -
categorical_accuracy: 0.6818 - precision: 0.7523 - recall: 0.5658 - f1_score:
0.6748 - val_loss: 0.9844 - val_categorical_accuracy: 0.6004 - val_precision:
0.6854 - val_recall: 0.4512 - val_f1_score: 0.5856
Epoch 6/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.7178 -
categorical_accuracy: 0.7088 - precision: 0.7658 - recall: 0.6166 - f1_score:
0.7032 - val_loss: 0.9385 - val_categorical_accuracy: 0.6260 - val_precision:
0.6775 - val_recall: 0.4460 - val_f1_score: 0.6029
Epoch 7/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.6823 -
categorical_accuracy: 0.7252 - precision: 0.7744 - recall: 0.6484 - f1_score:
0.7203 - val_loss: 0.8972 - val_categorical_accuracy: 0.6425 - val_precision:
0.6728 - val_recall: 0.5171 - val_f1_score: 0.6214
Epoch 8/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.6494 -
categorical_accuracy: 0.7414 - precision: 0.7848 - recall: 0.6772 - f1_score:
0.7370 - val_loss: 0.7888 - val_categorical_accuracy: 0.6900 - val_precision:
0.7273 - val_recall: 0.5767 - val_f1_score: 0.6780
```

```
Epoch 9/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.6203 -
categorical_accuracy: 0.7539 - precision: 0.7930 - recall: 0.6988 - f1_score:
0.7507 - val_loss: 0.8268 - val_categorical_accuracy: 0.6935 - val_precision:
0.7278 - val_recall: 0.5833 - val_f1_score: 0.6913
Epoch 10/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.6096 -
categorical_accuracy: 0.7576 - precision: 0.7975 - recall: 0.7064 - f1_score:
0.7547 - val_loss: 0.7561 - val_categorical_accuracy: 0.7119 - val_precision:
0.7499 - val_recall: 0.6171 - val_f1_score: 0.7101
Epoch 11/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.5797 -
categorical_accuracy: 0.7697 - precision: 0.8065 - recall: 0.7207 - f1_score:
0.7670 - val_loss: 0.8398 - val_categorical_accuracy: 0.6756 - val_precision:
0.7064 - val_recall: 0.5412 - val_f1_score: 0.6653
Epoch 12/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.5741 -
categorical_accuracy: 0.7749 - precision: 0.8102 - recall: 0.7275 - f1_score:
0.7726 - val_loss: 0.7955 - val_categorical_accuracy: 0.7215 - val_precision:
0.7685 - val_recall: 0.5817 - val_f1_score: 0.7218
Epoch 13/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.5523 -
categorical_accuracy: 0.7842 - precision: 0.8181 - recall: 0.7426 - f1_score:
0.7818 - val_loss: 0.7134 - val_categorical_accuracy: 0.7412 - val_precision:
0.7798 - val_recall: 0.6271 - val_f1_score: 0.7389
Epoch 14/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.5362 -
categorical_accuracy: 0.7906 - precision: 0.8231 - recall: 0.7511 - f1_score:
0.7886 - val_loss: 0.7583 - val_categorical_accuracy: 0.7292 - val_precision:
0.7745 - val_recall: 0.6313 - val_f1_score: 0.7287
Epoch 15/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.5250 -
categorical_accuracy: 0.7966 - precision: 0.8277 - recall: 0.7573 - f1_score:
0.7947 - val_loss: 0.7023 - val_categorical_accuracy: 0.7473 - val_precision:
0.7840 - val_recall: 0.6369 - val_f1_score: 0.7446
Epoch 16/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.5160 -
categorical_accuracy: 0.8057 - precision: 0.8354 - recall: 0.7684 - f1_score:
0.8042 - val_loss: 0.6940 - val_categorical_accuracy: 0.7594 - val_precision:
0.7918 - val_recall: 0.6687 - val_f1_score: 0.7594
Epoch 17/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.5065 -
categorical_accuracy: 0.8091 - precision: 0.8385 - recall: 0.7754 - f1_score:
0.8080 - val_loss: 0.6589 - val_categorical_accuracy: 0.7579 - val_precision:
0.8035 - val_recall: 0.6363 - val_f1_score: 0.7563
Epoch 18/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4979 -
categorical_accuracy: 0.8106 - precision: 0.8405 - recall: 0.7748 - f1_score:
```

```
0.8094 - val_loss: 0.6269 - val_categorical_accuracy: 0.7831 - val_precision:
0.8260 - val_recall: 0.6825 - val_f1_score: 0.7835
Epoch 19/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4862 -
categorical_accuracy: 0.8152 - precision: 0.8425 - recall: 0.7799 - f1_score:
0.8140 - val_loss: 0.6973 - val_categorical_accuracy: 0.7513 - val_precision:
0.7757 - val_recall: 0.6354 - val_f1_score: 0.7499
Epoch 20/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4921 -
categorical_accuracy: 0.8165 - precision: 0.8460 - recall: 0.7788 - f1_score:
0.8153 - val_loss: 0.6053 - val_categorical_accuracy: 0.7933 - val_precision:
0.8253 - val_recall: 0.7067 - val_f1_score: 0.7933
Epoch 21/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4702 -
categorical_accuracy: 0.8252 - precision: 0.8544 - recall: 0.7878 - f1_score:
0.8241 - val_loss: 0.6865 - val_categorical_accuracy: 0.7690 - val_precision:
0.8027 - val_recall: 0.6221 - val_f1_score: 0.7685
Epoch 22/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4659 -
categorical_accuracy: 0.8256 - precision: 0.8542 - recall: 0.7900 - f1_score:
0.8244 - val_loss: 0.6506 - val_categorical_accuracy: 0.7594 - val_precision:
0.7886 - val_recall: 0.6677 - val_f1_score: 0.7553
Epoch 23/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4642 -
categorical_accuracy: 0.8288 - precision: 0.8589 - recall: 0.7945 - f1_score:
0.8280 - val_loss: 0.6535 - val_categorical_accuracy: 0.7477 - val_precision:
0.8164 - val_recall: 0.6085 - val_f1_score: 0.7393
Epoch 24/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4438 -
categorical_accuracy: 0.8346 - precision: 0.8646 - recall: 0.8001 - f1_score:
0.8338 - val_loss: 0.6203 - val_categorical_accuracy: 0.7858 - val_precision:
0.8480 - val_recall: 0.6554 - val_f1_score: 0.7855
Epoch 25/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4364 -
categorical_accuracy: 0.8390 - precision: 0.8676 - recall: 0.8060 - f1_score:
0.8380 - val_loss: 0.5746 - val_categorical_accuracy: 0.8081 - val_precision:
0.8530 - val_recall: 0.7348 - val_f1_score: 0.8108
Epoch 26/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4375 -
categorical_accuracy: 0.8415 - precision: 0.8701 - recall: 0.8084 - f1_score:
0.8409 - val_loss: 0.5752 - val_categorical_accuracy: 0.8100 - val_precision:
0.8588 - val_recall: 0.7340 - val_f1_score: 0.8096
Epoch 27/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4462 -
categorical_accuracy: 0.8374 - precision: 0.8659 - recall: 0.8045 - f1_score:
0.8364 - val_loss: 0.7371 - val_categorical_accuracy: 0.7325 - val_precision:
0.7870 - val_recall: 0.6467 - val_f1_score: 0.7304
Epoch 28/75
```

```
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4468 -
categorical_accuracy: 0.8360 - precision: 0.8672 - recall: 0.8015 - f1_score:
0.8353 - val_loss: 0.5915 - val_categorical_accuracy: 0.8046 - val_precision:
0.8573 - val_recall: 0.6833 - val_f1_score: 0.8070
Epoch 29/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4280 -
categorical_accuracy: 0.8441 - precision: 0.8740 - recall: 0.8101 - f1_score:
0.8436 - val_loss: 0.5550 - val_categorical_accuracy: 0.8181 - val_precision:
0.8809 - val_recall: 0.7292 - val_f1_score: 0.8210
Epoch 30/75
1200/1200 [==============================] - 11s 9ms/step - loss: 0.4166 -
categorical_accuracy: 0.8491 - precision: 0.8776 - recall: 0.8168 - f1_score:
0.8482 - val_loss: 0.6458 - val_categorical_accuracy: 0.7925 - val_precision:
0.8573 - val_recall: 0.6321 - val_f1_score: 0.7957
Epoch 31/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4346 -
categorical_accuracy: 0.8439 - precision: 0.8725 - recall: 0.8105 - f1_score:
0.8429 - val_loss: 0.6460 - val_categorical_accuracy: 0.7531 - val_precision:
0.8142 - val_recall: 0.6656 - val_f1_score: 0.7544
Epoch 32/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4139 -
categorical_accuracy: 0.8516 - precision: 0.8782 - recall: 0.8204 - f1_score:
0.8511 - val_loss: 0.6485 - val_categorical_accuracy: 0.7590 - val_precision:
0.8298 - val_recall: 0.6529 - val_f1_score: 0.7601
Epoch 33/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4080 -
categorical_accuracy: 0.8528 - precision: 0.8813 - recall: 0.8229 - f1_score:
0.8525 - val_loss: 0.5358 - val_categorical_accuracy: 0.8252 - val_precision:
0.8622 - val_recall: 0.7590 - val_f1_score: 0.8273
Epoch 34/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4384 -
categorical_accuracy: 0.8468 - precision: 0.8776 - recall: 0.8117 - f1_score:
0.8462 - val_loss: 0.6149 - val_categorical_accuracy: 0.7827 - val_precision:
0.8646 - val_recall: 0.6452 - val_f1_score: 0.7840
Epoch 35/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4259 -
categorical_accuracy: 0.8479 - precision: 0.8786 - recall: 0.8139 - f1_score:
0.8471 - val_loss: 0.6066 - val_categorical_accuracy: 0.7867 - val_precision:
0.8451 - val_recall: 0.6946 - val_f1_score: 0.7888
Epoch 36/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.3983 -
categorical_accuracy: 0.8606 - precision: 0.8870 - recall: 0.8295 - f1_score:
0.8602 - val_loss: 0.5365 - val_categorical_accuracy: 0.8177 - val_precision:
0.8714 - val_recall: 0.7342 - val_f1_score: 0.8181
Epoch 37/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4023 -
categorical_accuracy: 0.8574 - precision: 0.8860 - recall: 0.8269 - f1_score:
0.8571 - val_loss: 0.5365 - val_categorical_accuracy: 0.8225 - val_precision:
```

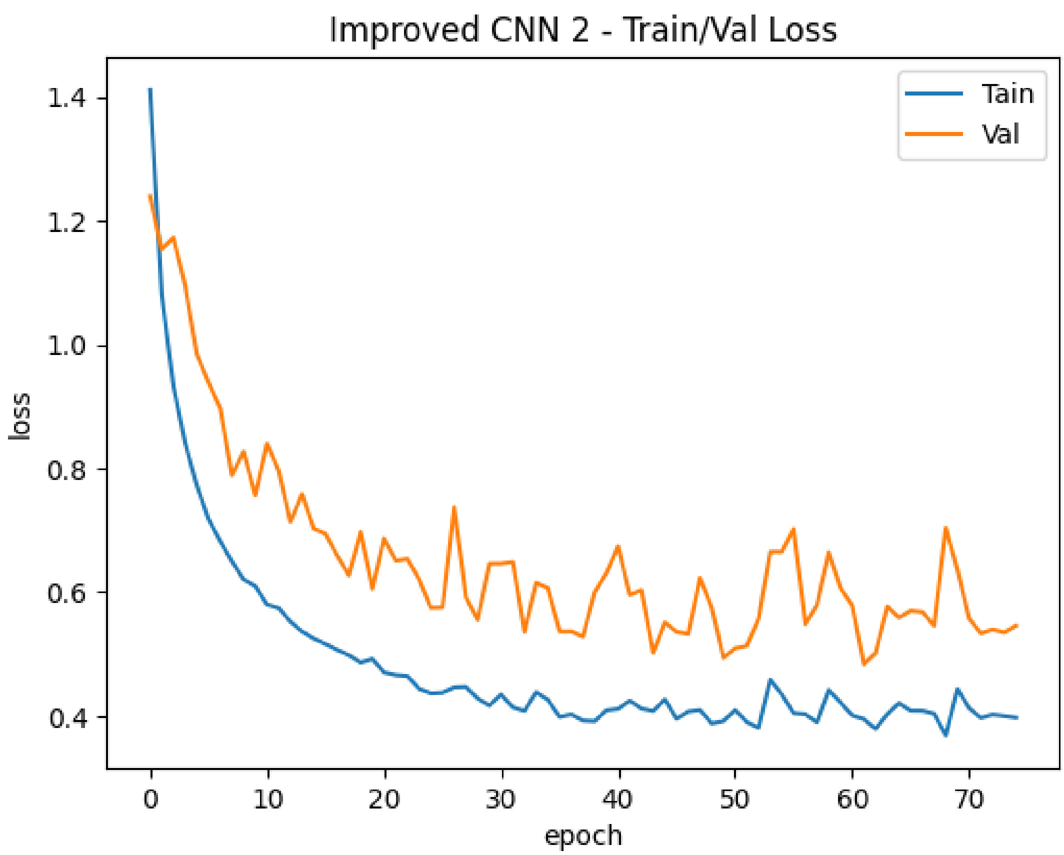
```
0.8768 - val_recall: 0.7429 - val_f1_score: 0.8243
Epoch 38/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3925 -
categorical_accuracy: 0.8633 - precision: 0.8892 - recall: 0.8351 - f1_score:
0.8629 - val_loss: 0.5280 - val_categorical_accuracy: 0.8217 - val_precision:
0.8741 - val_recall: 0.7408 - val_f1_score: 0.8219
Epoch 39/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3915 -
categorical_accuracy: 0.8611 - precision: 0.8878 - recall: 0.8309 - f1_score:
0.8607 - val_loss: 0.5999 - val_categorical_accuracy: 0.8002 - val_precision:
0.8553 - val_recall: 0.7167 - val_f1_score: 0.8030
Epoch 40/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4090 -
categorical_accuracy: 0.8561 - precision: 0.8829 - recall: 0.8258 - f1_score:
0.8557 - val_loss: 0.6305 - val_categorical_accuracy: 0.7748 - val_precision:
0.8504 - val_recall: 0.6929 - val_f1_score: 0.7750
Epoch 41/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4114 -
categorical_accuracy: 0.8584 - precision: 0.8866 - recall: 0.8295 - f1_score:
0.8581 - val_loss: 0.6739 - val_categorical_accuracy: 0.7465 - val_precision:
0.8060 - val_recall: 0.6258 - val_f1_score: 0.7452
Epoch 42/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4243 -
categorical_accuracy: 0.8545 - precision: 0.8822 - recall: 0.8207 - f1_score:
0.8544 - val_loss: 0.5955 - val_categorical_accuracy: 0.7823 - val_precision:
0.8495 - val_recall: 0.7044 - val_f1_score: 0.7827
Epoch 43/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4118 -
categorical_accuracy: 0.8569 - precision: 0.8837 - recall: 0.8254 - f1_score:
0.8567 - val_loss: 0.6031 - val_categorical_accuracy: 0.7894 - val_precision:
0.8797 - val_recall: 0.6490 - val_f1_score: 0.7932
Epoch 44/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4079 -
categorical_accuracy: 0.8577 - precision: 0.8842 - recall: 0.8256 - f1_score:
0.8574 - val_loss: 0.5019 - val_categorical_accuracy: 0.8479 - val_precision:
0.9030 - val_recall: 0.7448 - val_f1_score: 0.8480
Epoch 45/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4265 -
categorical_accuracy: 0.8546 - precision: 0.8827 - recall: 0.8209 - f1_score:
0.8544 - val_loss: 0.5508 - val_categorical_accuracy: 0.8110 - val_precision:
0.8739 - val_recall: 0.7292 - val_f1_score: 0.8138
Epoch 46/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.3953 -
categorical_accuracy: 0.8616 - precision: 0.8896 - recall: 0.8302 - f1_score:
0.8612 - val_loss: 0.5358 - val_categorical_accuracy: 0.8273 - val_precision:
0.8673 - val_recall: 0.7573 - val_f1_score: 0.8287
Epoch 47/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4070 -
```

```
categorical_accuracy: 0.8557 - precision: 0.8856 - recall: 0.8255 - f1_score:
0.8553 - val_loss: 0.5324 - val_categorical_accuracy: 0.8210 - val_precision:
0.8714 - val_recall: 0.7415 - val_f1_score: 0.8211
Epoch 48/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4095 -
categorical_accuracy: 0.8537 - precision: 0.8855 - recall: 0.8192 - f1_score:
0.8531 - val_loss: 0.6228 - val_categorical_accuracy: 0.7802 - val_precision:
0.8705 - val_recall: 0.6315 - val_f1_score: 0.7815
Epoch 49/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.3877 -
categorical_accuracy: 0.8645 - precision: 0.8935 - recall: 0.8308 - f1_score:
0.8639 - val_loss: 0.5747 - val_categorical_accuracy: 0.8023 - val_precision:
0.8730 - val_recall: 0.7046 - val_f1_score: 0.8029
Epoch 50/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.3916 -
categorical_accuracy: 0.8670 - precision: 0.8959 - recall: 0.8339 - f1_score:
0.8668 - val_loss: 0.4937 - val_categorical_accuracy: 0.8360 - val_precision:
0.9029 - val_recall: 0.7577 - val_f1_score: 0.8370
Epoch 51/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4094 -
categorical_accuracy: 0.8590 - precision: 0.8910 - recall: 0.8232 - f1_score:
0.8589 - val_loss: 0.5091 - val_categorical_accuracy: 0.8273 - val_precision:
0.8968 - val_recall: 0.7348 - val_f1_score: 0.8274
Epoch 52/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3901 -
categorical_accuracy: 0.8645 - precision: 0.8955 - recall: 0.8304 - f1_score:
0.8643 - val_loss: 0.5130 - val_categorical_accuracy: 0.8246 - val_precision:
0.8931 - val_recall: 0.7294 - val_f1_score: 0.8281
Epoch 53/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3804 -
categorical_accuracy: 0.8684 - precision: 0.8962 - recall: 0.8390 - f1_score:
0.8683 - val_loss: 0.5569 - val_categorical_accuracy: 0.8175 - val_precision:
0.8791 - val_recall: 0.7244 - val_f1_score: 0.8181
Epoch 54/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4586 -
categorical_accuracy: 0.8481 - precision: 0.8806 - recall: 0.8027 - f1_score:
0.8475 - val_loss: 0.6650 - val_categorical_accuracy: 0.7721 - val_precision:
0.8520 - val_recall: 0.6633 - val_f1_score: 0.7715
Epoch 55/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4343 -
categorical_accuracy: 0.8528 - precision: 0.8863 - recall: 0.8125 - f1_score:
0.8527 - val_loss: 0.6652 - val_categorical_accuracy: 0.7175 - val_precision:
0.9013 - val_recall: 0.5460 - val_f1_score: 0.7040
Epoch 56/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4043 -
categorical_accuracy: 0.8609 - precision: 0.8929 - recall: 0.8233 - f1_score:
0.8606 - val_loss: 0.7021 - val_categorical_accuracy: 0.7081 - val_precision:
0.8444 - val_recall: 0.5835 - val_f1_score: 0.7108
```

```
Epoch 57/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4026 -
categorical_accuracy: 0.8639 - precision: 0.8932 - recall: 0.8295 - f1_score:
0.8634 - val_loss: 0.5480 - val_categorical_accuracy: 0.8208 - val_precision:
0.8703 - val_recall: 0.7158 - val_f1_score: 0.8228
Epoch 58/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3895 -
categorical_accuracy: 0.8695 - precision: 0.8964 - recall: 0.8380 - f1_score:
0.8692 - val_loss: 0.5794 - val_categorical_accuracy: 0.8167 - val_precision:
0.8781 - val_recall: 0.6977 - val_f1_score: 0.8194
Epoch 59/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4420 -
categorical_accuracy: 0.8529 - precision: 0.8856 - recall: 0.8127 - f1_score:
0.8528 - val_loss: 0.6646 - val_categorical_accuracy: 0.7333 - val_precision:
0.8691 - val_recall: 0.6169 - val_f1_score: 0.7304
Epoch 60/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4216 -
categorical_accuracy: 0.8594 - precision: 0.8912 - recall: 0.8232 - f1_score:
0.8595 - val_loss: 0.6068 - val_categorical_accuracy: 0.7956 - val_precision:
0.8657 - val_recall: 0.6742 - val_f1_score: 0.7996
Epoch 61/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4008 -
categorical_accuracy: 0.8657 - precision: 0.8954 - recall: 0.8322 - f1_score:
0.8659 - val_loss: 0.5783 - val_categorical_accuracy: 0.8033 - val_precision:
0.8713 - val_recall: 0.7138 - val_f1_score: 0.8075
Epoch 62/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3947 -
categorical_accuracy: 0.8675 - precision: 0.8979 - recall: 0.8326 - f1_score:
0.8681 - val_loss: 0.4835 - val_categorical_accuracy: 0.8406 - val_precision:
0.8964 - val_recall: 0.7588 - val_f1_score: 0.8426
Epoch 63/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.3790 -
categorical_accuracy: 0.8711 - precision: 0.9004 - recall: 0.8383 - f1_score:
0.8716 - val_loss: 0.5013 - val_categorical_accuracy: 0.8481 - val_precision:
0.9020 - val_recall: 0.7496 - val_f1_score: 0.8491
Epoch 64/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4028 -
categorical_accuracy: 0.8626 - precision: 0.8932 - recall: 0.8269 - f1_score:
0.8625 - val_loss: 0.5768 - val_categorical_accuracy: 0.7887 - val_precision:
0.8786 - val_recall: 0.6888 - val_f1_score: 0.7911
Epoch 65/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4205 -
categorical_accuracy: 0.8636 - precision: 0.8959 - recall: 0.8237 - f1_score:
0.8634 - val_loss: 0.5586 - val_categorical_accuracy: 0.8148 - val_precision:
0.8901 - val_recall: 0.7069 - val_f1_score: 0.8176
Epoch 66/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4084 -
categorical_accuracy: 0.8645 - precision: 0.8956 - recall: 0.8266 - f1_score:
```

```
0.8643 - val_loss: 0.5698 - val_categorical_accuracy: 0.8156 - val_precision:
0.8716 - val_recall: 0.7113 - val_f1_score: 0.8165
Epoch 67/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4083 -
categorical_accuracy: 0.8627 - precision: 0.8948 - recall: 0.8256 - f1_score:
0.8628 - val_loss: 0.5677 - val_categorical_accuracy: 0.8206 - val_precision:
0.8946 - val_recall: 0.7073 - val_f1_score: 0.8237
Epoch 68/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4035 -
categorical_accuracy: 0.8648 - precision: 0.8959 - recall: 0.8266 - f1_score:
0.8648 - val_loss: 0.5456 - val_categorical_accuracy: 0.8167 - val_precision:
0.8884 - val_recall: 0.7231 - val_f1_score: 0.8204
Epoch 69/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3686 -
categorical_accuracy: 0.8763 - precision: 0.9039 - recall: 0.8461 - f1_score:
0.8763 - val_loss: 0.7036 - val_categorical_accuracy: 0.7258 - val_precision:
0.8396 - val_recall: 0.6313 - val_f1_score: 0.7161
Epoch 70/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4433 -
categorical_accuracy: 0.8525 - precision: 0.8861 - recall: 0.8172 - f1_score:
0.8531 - val_loss: 0.6355 - val_categorical_accuracy: 0.7796 - val_precision:
0.8573 - val_recall: 0.6623 - val_f1_score: 0.7773
Epoch 71/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.4124 -
categorical_accuracy: 0.8622 - precision: 0.8925 - recall: 0.8295 - f1_score:
0.8620 - val_loss: 0.5570 - val_categorical_accuracy: 0.8292 - val_precision:
0.8922 - val_recall: 0.7204 - val_f1_score: 0.8305
Epoch 72/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.3968 -
categorical_accuracy: 0.8709 - precision: 0.8985 - recall: 0.8386 - f1_score:
0.8709 - val_loss: 0.5334 - val_categorical_accuracy: 0.8319 - val_precision:
0.8738 - val_recall: 0.7412 - val_f1_score: 0.8353
Epoch 73/75
1200/1200 [==============================] - 10s 8ms/step - loss: 0.4021 -
categorical_accuracy: 0.8679 - precision: 0.8992 - recall: 0.8338 - f1_score:
0.8681 - val_loss: 0.5394 - val_categorical_accuracy: 0.8267 - val_precision:
0.8783 - val_recall: 0.7456 - val_f1_score: 0.8271
Epoch 74/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3998 -
categorical_accuracy: 0.8660 - precision: 0.8948 - recall: 0.8342 - f1_score:
0.8659 - val_loss: 0.5348 - val_categorical_accuracy: 0.8404 - val_precision:
0.9026 - val_recall: 0.7127 - val_f1_score: 0.8425
Epoch 75/75
1200/1200 [==============================] - 10s 9ms/step - loss: 0.3971 -
categorical_accuracy: 0.8721 - precision: 0.9041 - recall: 0.8385 - f1_score:
0.8723 - val_loss: 0.5456 - val_categorical_accuracy: 0.8240 - val_precision:
0.8838 - val_recall: 0.7398 - val_f1_score: 0.8257
```

[ ]:
```
#plot loss
plt.plot(history_cnn_exp2.history['loss'])
plt.plot(history_cnn_exp2.history['val_loss'])
plt.title('Improved CNN 2 - Train/Val Loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['Tain', 'Val'])
plt.show()
```



[ ]:
```
# evaluate on val data
loss, accuracy, precision, recall, f1 = cnn_exp2.evaluate(X_val, y_val)
print(f'Val Loss: {loss}\nAccuracy: {accuracy}\nPrecision: {precision}\nRecall:␣
↪{recall},\nF1: {f1}')
```

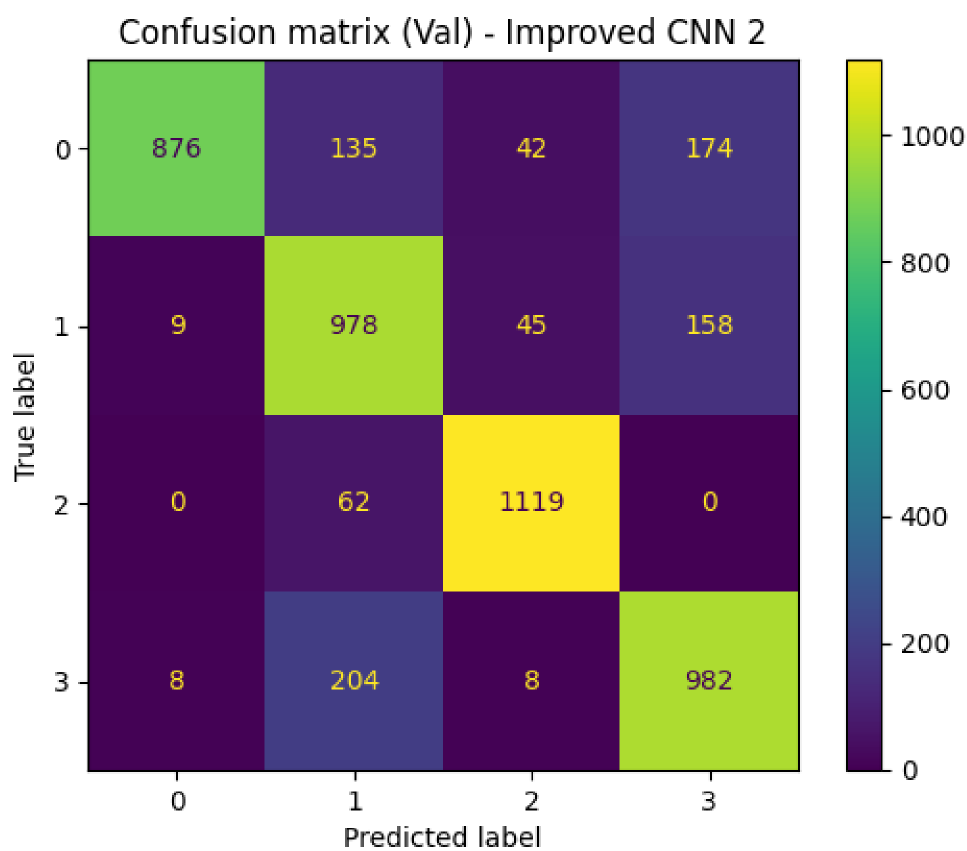
```
150/150 [==============================] - 1s 5ms/step - loss: 0.5456 -
categorical_accuracy: 0.8240 - precision: 0.8838 - recall: 0.7398 - f1_score:
0.8257
Val Loss: 0.5456002354621887
Accuracy: 0.8239583373069763
Precision: 0.8837730288505554
```

```
Recall: 0.7397916913032532,
F1: [0.82641506 0.76138574 0.9344468  0.78060406]
```

[ ]:
```python
# plot confusion matrix
y_pred_cnn2 = cnn_exp2.predict(X_val)
cm = confusion_matrix(np.argmax(y_val, axis=1), np.argmax(y_pred_cnn2, axis=1))
ConfusionMatrixDisplay(confusion_matrix=cm).plot();
plt.title('Confusion matrix (Val) - Improved CNN 2')
```

```
150/150 [==============================] - 1s 4ms/step
```

[ ]:
```
Text(0.5, 1.0, 'Confusion matrix (Val) - Improved CNN 2')
```



1.5 Evaluate Best Model on Test Dataset

We will select our best performing model vs. the validation set and now evaluate performance against our hold-out test dataset. The model will be evaluated with metrics Categorical Accuracy, Precision, Recall and F1 Score.

[ ]:
```python
# load dataset
X_train, y_train, X_val, y_val, X_test, y_test = load_prepared_data()
```

[ ]:
```python
# load out best CNN model
best_cnn = tf.keras.models.load_model('/content/drive/MyDrive/USD/models/
 ↪composer-classifier/cnn-2-2')
```

[ ]:
```python
# confirm model
best_cnn.summary()
```

```
Model: "sequential"

_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 normalization (Normalizati  (None, 200, 128)          3
 on)

 conv1d (Conv1D)             (None, 200, 512)          197120

 dropout (Dropout)           (None, 200, 512)          0

 max_pooling1d (MaxPooling1  (None, 100, 512)          0
 D)

 conv1d_1 (Conv1D)           (None, 100, 256)          393472

 dropout_1 (Dropout)         (None, 100, 256)          0

 max_pooling1d_1 (MaxPoolin  (None, 50, 256)           0
 g1D)

 conv1d_2 (Conv1D)           (None, 50, 128)           98432

 dropout_2 (Dropout)         (None, 50, 128)           0

 max_pooling1d_2 (MaxPoolin  (None, 25, 128)           0
 g1D)

 conv1d_3 (Conv1D)           (None, 25, 64)            24640

 dropout_3 (Dropout)         (None, 25, 64)            0

 max_pooling1d_3 (MaxPoolin  (None, 12, 64)            0
 g1D)

 conv1d_4 (Conv1D)           (None, 12, 32)            6176

 dropout_4 (Dropout)         (None, 12, 32)            0

 max_pooling1d_4 (MaxPoolin  (None, 6, 32)             0
 g1D)
```

```
 flatten (Flatten)           (None, 192)               0

 dense (Dense)               (None, 64)                12352

 dropout_5 (Dropout)         (None, 64)                0

 dense_1 (Dense)             (None, 64)                4160

 dropout_6 (Dropout)         (None, 64)                0

 dense_2 (Dense)             (None, 32)                2080

 dropout_7 (Dropout)         (None, 32)                0

 dense_3 (Dense)             (None, 4)                 132

=================================================================
Total params: 738567 (2.82 MB)
Trainable params: 738564 (2.82 MB)
Non-trainable params: 3 (16.00 Byte)
_________________________________________________________________
```

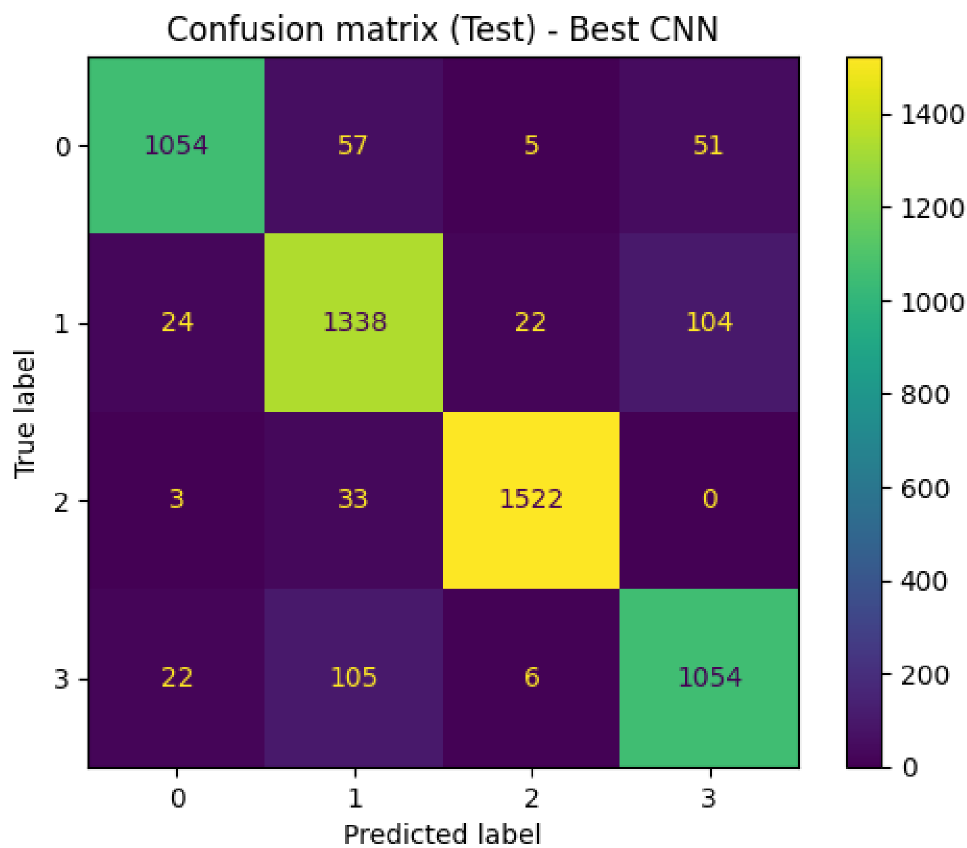
[ ]:
```
# run eval on test dataset
loss, accuracy, precision, recall, f1 = best_cnn.evaluate(X_test, y_test)
print(f'Test Loss: {loss}\nAccuracy: {accuracy}\nPrecision: {precision}\nRecall:
 ↪ {recall},\nF1: {f1}')
```

```
169/169 [==============================] - 5s 24ms/step - loss: 0.2803 -
categorical_accuracy: 0.9200 - precision: 0.9312 - recall: 0.9093 - f1_score:
0.9180
Test Loss: 0.2803073525428772
Accuracy: 0.9200000166893005
Precision: 0.931158721446991
Recall: 0.9092592597007751,
F1: [0.92863435 0.8857994  0.9778349  0.8797996 ]
```

[ ]:
```
# plot confusion matrix
y_pred_test = best_cnn.predict(X_test)
cm = confusion_matrix(np.argmax(y_test, axis=1), np.argmax(y_pred_test, axis=1))
ConfusionMatrixDisplay(confusion_matrix=cm).plot();
plt.title('Confusion matrix (Test) - Best CNN')
```

```
169/169 [==============================] - 4s 25ms/step
```

[ ]:
```
Text(0.5, 1.0, 'Confusion matrix (Test) - Best CNN')
```


Confusion matrix (Test) - Best CNN
1054
57
5
51
24
1338
22
104
3
33
1522
0
22
105
6
1054
True label
Predicted label
0
1
2
3
0
1
2
3
1400
1200
1000
800
600
400
200
0