Loading Dependencies:

Mounting Google Drive:

In [2]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [4]:
```
# for accessing tabular data
import pandas as pd
import numpy as np
import os
os.chdir('/content/drive/My Drive/')
# adding classweight
from sklearn.utils import class_weight
# Evaluation Metric
from sklearn.metrics import cohen_kappa_score
from sklearn.metrics import confusion_matrix, precision_score, recall_score
# for visualization
import cv2
import matplotlib.pyplot as plt
import seaborn as sns
from prettytable import PrettyTable
# backend
import keras
from keras import backend as K
import tensorflow as tf
from keras.callbacks import Callback
# for transfer learning
from keras.applications import VGG16, VGG19
from keras.applications import DenseNet121
from keras.applications import ResNet50, ResNet152
from keras.applications import InceptionV3
from efficientnet.keras import EfficientNetB0, EfficientNetB3, EfficientNetB4
from keras.applications import Xception
# for model architecture
from keras.models import Sequential
from keras.layers import GlobalAveragePooling2D, Dropout, Dense, Conv2D, MaxPooling2D, Activation, Flatten
# for Tensorboard visualization
from keras.callbacks import TensorBoard
# for Data Augmentation
from keras.preprocessing.image import ImageDataGenerator
```

In [ ]:
```
try:
    x_train = np.load("processed_images/training.npy", mmap_mode=None, allow_pickle=False, fix_imports=True)
    x_validation = np.load("processed_images/validation.npy",mmap_mode=None, allow_pickle=False, fix_imports=True)
    x_test = np.load("processed_images/test.npy", mmap_mode = None,allow_pickle = False, fix_imports = True)
    print("Loaded Successfully...\n")
    print(x_train.shape)
    print(x_validation.shape)
    print(x_test.shape)
except:
    print("file not exists")
```

```
Loaded Successfully...

(3112, 512, 512, 3)
(550, 512, 512, 3)
(1928, 512, 512, 3)
```

In [ ]:
```
train_labels = pd.read_csv('training.csv')
train_labels = train_labels['diagnosis']
validation_labels = pd.read_csv('validation.csv')
validation_labels = validation_labels['diagnosis']
print("Training:",train_labels.shape[0])
print("Validation:",validation_labels.shape[0])
```

```
Training: 3112
Validation: 550
```

Transforming Target Labels:

There are five outcomes in our model i.e., [0,1,2,3,4]. Now we can transform these labels either using

1. **One hot encoding**
   For example if the target label is 2 then it will encoded as [0,0,1,0,0].
2. **Ordinal Regression**
   For example if the target label is 2 then it will encoded as [1,1,1,0,0] which means the sample with class-2 also belongs to the classes before it like 0 and 1(here)

According to the this paper (https://arxiv.org/pdf/0704.1028.pdf) using ordinal regression especially in case of categorical target variables that too in healthcare will be of more helpful.

In [ ]:
```
def ordinal_encoding(labels):
    """
    This function is used to create one_hot_encoding of the labels.
    E.x: category = 3 -> one-hot-encoding [1,1,1,1,0].
         category = 2 -> one-hot-encoding [1,1,1,0,0].
    """
    y_train = pd.get_dummies(labels).values
    y_train_multi = np.empty(y_train.shape, dtype=y_train.dtype)
    y_train_multi[:, 4] = y_train[:, 4]

    for i in range(3, -1, -1):
         y_train_multi[:, i] = np.logical_or(y_train[:, i], y_train_multi[:, i+1])
    return y_train_multi
labels_train = ordinal_regression(train_labels)
labels_validation = ordinal_regression(validation_labels)
print(labels_train.shape)
print(labels_validation.shape)
print("Ex: Original Category: {}\n After one_hot_encoding: {}".format(train_labels.iloc[23],labels_train[23]))
```

```
(3112, 5)
(550, 5)
Ex: Original Category: 0
 After one_hot_encoding: [1 0 0 0 0]
```

Computing Classweight:

Since the data set is highly imbalanced adding classweight can make it balanced by adding weights to each of the classes.
The class with more weight means they are less in samples. Similarly the class with less weight means they are more in samples.

In [ ]:
```
class_weights = class_weight.compute_class_weight('balanced',[0,1,2,3,4],train_labels)
print(class_weights)
```

```
[0.40573664 1.98216561 0.73309776 3.79512195 2.47968127]
```

Cohen's Kappa:

- It is used to measure the degree of agreement between raters for categorical data.
- It is an extension to accuracy where it finds the simple percent agreement calculation (True predictions / Total predictions). Kappa score makes it more robust by considering the agreement occured by chance.

$$\kappa \equiv \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e},$$

where,
Po is observed agreement (same as accuracy).
Pe is chance of disagreement observed between raters.

Quadratic Weighted Cohen's kappa Score:

- The weighted kappa allows disagreements to be weighted differently and it is usefully when codes are ordered.
- It makes use of three matrices.
  1. The matrix of observed scores.
  2. The matrix of expected scores.
  3. Weight matrix

$$\kappa = 1 - \frac{\sum_{i=1}^{k} \sum_{j=1}^{k} w_{ij} x_{ij}}{\sum_{i=1}^{k} \sum_{j=1}^{k} w_{ij} m_{ij}}$$

where,
wij is elements in weight matrix
xij is elements in observed matrix
mij is elements in expected matrix

Why using weighted kappa metric??

- It is usefull when the target labels are ordered.
- In case of disagreement, the score is given according to the distance of predicted and observed elements. It is not only taking care of agreement but also posing penality on the disagreement.
- The score will be higher if the true prediction is 3 and model predicted as 2 and will be lower if the model prediction is 4. This property will be very helpful because in medical getting higher than expected will be okay because on later diagnosis it can be rectified. But if the patient got class-1 and it is predicted as class-0 then it will be a problem because that patient is ignored which can adversly effect the patient as well as the reputation of the hospital.

Read more about kappa score here (https://en.wikipedia.org/wiki/Cohen%27s_kappa#Weighted_kappa) and here (https://www.kaggle.com/aroraaman/quadratic-kappa-metric-explained-in-5-simple-steps).

In [ ]:
```
def kappa_metric(y_true, y_pred):
    """
    This function is used for calculating kappa score between the model predicted values and true values
    Args: y_true (np.ndarray or list) - true values
          y_pred (np.ndarray or list) - model predicted value
    Output: _kappa_ (integer or float) - return's kappa score
    """
    y_true = y_true.sum(axis=1) - 1
    y_pred = y_pred > 0.5
    y_pred = y_pred.astype(int).sum(axis=1) - 1
    _kappa_ = cohen_kappa_score( y_true, y_pred,  weights='quadratic' )
    return _kappa_
```

In [ ]:
```
class Metrics(Callback):
    def __init__(self, path):
        ''' Initialization of variables '''
        super(Callback, self).__init__()
        self.path = path
    def ModelCheckPoint(self, path):
        ''' This function is used for Storing the model weights if val_kappa improves from all the previous epochs'''
        self.model.save(path)
    def on_train_begin(self, logs={}):
        self.val_kappas = []
    def on_epoch_end(self, epoch, logs={}):
        ''' This function is used for calculating kappa score on each epoch and updates the validation kappa score if it improves from the previous epochs'''
        X_val, y_val = self.validation_data[:2]
        y_val = y_val.sum(axis=1) - 1
        y_pred = self.model.predict(X_val) > 0.5
        y_pred = y_pred.astype(int).sum(axis=1) - 1
        _val_kappa = cohen_kappa_score(
            y_val,
            y_pred,
            weights='quadratic'
        )
        self.val_kappas.append(_val_kappa)
        print(f"\b - val_kappa:  {_val_kappa:.4f}")
        if _val_kappa == max(self.val_kappas):
            print("\n\t\tValidation Kappa has improved. Saving model to {}...".format(self.path))
            self.ModelCheckPoint(self.path)
        else:
            print("\n\t\tValidation kappa did not improved from {}".format(max(self.val_kappas)))
        return
```

In [ ]:
```
class PerformanceMetric:
    def __init__(self,actual_labels,predicted_labels):
        ''' Initialization of variables '''
        self.actual_labels = actual_labels
        self.predicted_labels = predicted_labels
    def single_value_conversion(self):
        ''' This function is used for Converting model predicted values into single values
           Ex: model_predicted_value: [0,1,0,0,0] and it converts as 1'''
        predicted_labels = self.predicted_labels > 0.5
        prediction_ordinal = np.empty(predicted_labels.shape, dtype = int)
        prediction_ordinal[:,4] = predicted_labels[:,4]
        for i in range(3, -1, -1): prediction_ordinal[:, i] = np.logical_or(predicted_labels[:,i], prediction_ordinal[:,i+1])
        self.predicted_labels = prediction_ordinal.sum(axis = 1)-1
        self.actual_labels = self.actual_labels.sum(axis = 1)-1
    def confusionMatrix(self):
        ''' This function is used for calculating confusion matrix between model predicted values and true values using sklearn implementation.'''
        confusion_matrix_ = confusion_matrix(self.actual_labels, self.predicted_labels)
        return confusion_matrix_
    def precision(self, matrix):
        ''' This function is used for calculating precision matrix between predicted values and true values using confusion matrix'''
        precision_matrix =(((matrix.T)/(matrix.sum(axis=1))).T)
        return precision_matrix
    def recall(self, matrix):
        ''' this function is used for calculating recall matrix between predicted values and true values using confusion matrix'''
        recall_matrix =(matrix/matrix.sum(axis=0))
        return recall_matrix
    def subplot_(self, matrix, i, title):
        ''' This function is used for subplots'''
        plt.subplot(1,3,i)
        labels = [1,2,3,4,5]
        sns.heatmap(matrix, annot=True, cmap=sns.light_palette('green'),linewidths = 0.8,cbar = False, fmt=".3f", xticklabels=labels, yticklabels=labels)
        plt.title(title)
        plt.xlabel('Predicted Class Labels')
        plt.ylabel('Actual Class Labels')
    def plotting(self):
        """
        This function is used for calculating number of misclassified points, confusion, recall and precision matrixes and plotting it using subplots.
        """
        self.single_value_conversion()
        confusion_matrix = self.confusionMatrix()
        #print("Number of misclassified points: ",(len(self.actual_labels)-np.trace(confusion_matrix))/len(self.actual_labels)*100,"\n")
        precision_matrix = self.precision(confusion_matrix)
        recall_matrix = self.recall(confusion_matrix)
        plt.figure(figsize=(20,5))
        self.subplot_(confusion_matrix, 1, 'Confusion Matrix')
        self.subplot_(precision_matrix, 2, 'Precision')
        self.subplot_(recall_matrix, 3, 'Recall')
        plt.show()
```

In [ ]:
```
def plotting(iter, val_kappa):
    """
    This function is used for plotting validation_kappa on each epoch
    Args: iter(integer) - Number of epochs we runned on model training.
          val_kappa(list) - validation kappa score on each epoch
    Output: None - It doesn't return anything.
    """
    epoch = [i for i in range(iter)]
    plt.plot(epoch,val_kappa)
    plt.title('validation_kappa on each epoch')
    plt.xlabel('epoch')
    plt.ylabel('val_kappa')
    plt.grid()
    plt.show()
```

In [ ]:
```
def test_prediction(predicted_labels):
    '''
    Making predictions of the probability scores. The class with more score will be taken as predicted class.
    Arguments:
    predicted_labels - (np.array) - probability score of given sample
    '''
    predicted_labels = predicted_labels > 0.5
    prediction_ordinal = np.empty(predicted_labels.shape, dtype = int)
    prediction_ordinal[:,4] = predicted_labels[:,4]
    for i in range(3, -1, -1): prediction_ordinal[:, i] = np.logical_or(predicted_labels[:,i], prediction_ordinal[:,i+1])
    predicted_labels = prediction_ordinal.sum(axis = 1)-1
    return predicted_labels
```

Model-Training:

Baseline Model:

First we are starting with a basemodel to estimate the final score. Here we are using

1. 3 Convolutional layers with fixed kernel size of 2x2 and padding = same with relu activation
2. 3 Maxpooling layers with fixed kernel size of 2x2.
3. Global average pooling after max pooling.
4. One dropout layer.
5. 4 Dense layers.

In [ ]:
```
def baseline_model():
    ''' This function is used for building a base line convolutional neural network architecture '''
    model = Sequential()
    model.add(Conv2D(filters=16, kernel_size=(2, 2), input_shape=[512,512,3], activation= 'relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Conv2D(filters=32, kernel_size=(2, 2), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Conv2D(filters=64, kernel_size=(2, 2), activation='relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(GlobalAveragePooling2D())
    model.add(Dense(units=128, activation = 'relu'))
    model.add(Dense(units=256, activation = 'relu'))
    model.add(Dropout(rate=0.2))
    model.add(Dense(units=512, activation='relu'))
    model.add(Dense(5, activation='sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
baseline = baseline_model()
baseline.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 511, 511, 16)      208
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 255, 255, 16)      0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 254, 254, 32)      2080
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 127, 127, 32)      0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 126, 126, 64)      8256
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 63, 63, 64)        0
_________________________________________________________________
global_average_pooling2d_1 ( (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 128)               8320
_________________________________________________________________
dense_5 (Dense)              (None, 256)               33024
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_6 (Dense)              (None, 512)               131584
_________________________________________________________________
dense_7 (Dense)              (None, 5)                 2565
=================================================================
Total params: 186,037
Trainable params: 186,037
Non-trainable params: 0
_________________________________________________________________
```

In [ ]:
```
baseline = baseline_model()
kappa_metrics = Metrics('/content/drive/My Drive/models/baseline1.h5')
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/baseline1')
call_backs = [kappa_metrics,tensorboard]
history = baseline.fit(x_train, labels_train, epochs = 30, batch_size = 32, verbose = 2, class_weight = class_weight, validation_data = (x_validation, labels_validation), callbacks = call_backs)
```

```
Train on 3112 samples, validate on 550 samples
Epoch 1/30

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (0.116181). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

 - 3s - loss: 1.0752 - accuracy: 0.7190 - val_loss: 0.4214 - val_accuracy: 0.7829
 - val_kappa:  0.0901

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 2/30
 - 2s - loss: 0.5839 - accuracy: 0.7546 - val_loss: 0.4111 - val_accuracy: 0.7749
 - val_kappa:  0.0000

Validation kappa did not improved from 0.09013247055438767
Epoch 3/30
 - 3s - loss: 0.4887 - accuracy: 0.7662 - val_loss: 0.3937 - val_accuracy: 0.8033
 - val_kappa:  0.1947

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 4/30
 - 2s - loss: 0.4312 - accuracy: 0.7883 - val_loss: 0.3862 - val_accuracy: 0.7938
 - val_kappa:  0.1235

Validation kappa did not improved from 0.19468523950904004
Epoch 5/30
 - 2s - loss: 0.4126 - accuracy: 0.8009 - val_loss: 0.3741 - val_accuracy: 0.8451
 - val_kappa:  0.4352

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 6/30
 - 2s - loss: 0.3906 - accuracy: 0.8164 - val_loss: 0.3613 - val_accuracy: 0.8502
 - val_kappa:  0.4263

Validation kappa did not improved from 0.4352454246980735
Epoch 7/30
 - 2s - loss: 0.3785 - accuracy: 0.8222 - val_loss: 0.3794 - val_accuracy: 0.7993
 - val_kappa:  0.1122

Validation kappa did not improved from 0.4352454246980735
Epoch 8/30
 - 2s - loss: 0.3749 - accuracy: 0.8253 - val_loss: 0.3728 - val_accuracy: 0.8069
 - val_kappa:  0.1463

Validation kappa did not improved from 0.4352454246980735
Epoch 9/30
 - 2s - loss: 0.3596 - accuracy: 0.8403 - val_loss: 0.3460 - val_accuracy: 0.8593
 - val_kappa:  0.4922

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 10/30
 - 2s - loss: 0.3568 - accuracy: 0.8379 - val_loss: 0.3357 - val_accuracy: 0.8593
 - val_kappa:  0.4799

Validation kappa did not improved from 0.4922454587475771
Epoch 11/30
 - 2s - loss: 0.3418 - accuracy: 0.8489 - val_loss: 0.3278 - val_accuracy: 0.8600
 - val_kappa:  0.4843

Validation kappa did not improved from 0.4922454587475771
Epoch 12/30
 - 2s - loss: 0.3318 - accuracy: 0.8562 - val_loss: 0.3235 - val_accuracy: 0.8538
 - val_kappa:  0.4095

Validation kappa did not improved from 0.4922454587475771
Epoch 13/30
 - 2s - loss: 0.3313 - accuracy: 0.8540 - val_loss: 0.3436 - val_accuracy: 0.8309
 - val_kappa:  0.2662

Validation kappa did not improved from 0.4922454587475771
Epoch 14/30
 - 2s - loss: 0.3202 - accuracy: 0.8585 - val_loss: 0.3161 - val_accuracy: 0.8567
 - val_kappa:  0.4233

Validation kappa did not improved from 0.4922454587475771
Epoch 15/30
 - 2s - loss: 0.3150 - accuracy: 0.8656 - val_loss: 0.3023 - val_accuracy: 0.8665
 - val_kappa:  0.4944

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 16/30
 - 2s - loss: 0.3056 - accuracy: 0.8704 - val_loss: 0.2930 - val_accuracy: 0.8742
 - val_kappa:  0.5432

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 17/30
 - 3s - loss: 0.2991 - accuracy: 0.8687 - val_loss: 0.2907 - val_accuracy: 0.8764
 - val_kappa:  0.5536

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 18/30
 - 3s - loss: 0.2964 - accuracy: 0.8705 - val_loss: 0.2905 - val_accuracy: 0.8742
 - val_kappa:  0.5675

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 19/30
 - 3s - loss: 0.2850 - accuracy: 0.8761 - val_loss: 0.2767 - val_accuracy: 0.8800
 - val_kappa:  0.5829

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 20/30
 - 2s - loss: 0.2842 - accuracy: 0.8785 - val_loss: 0.2780 - val_accuracy: 0.8804
 - val_kappa:  0.5848

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 21/30
 - 2s - loss: 0.2749 - accuracy: 0.8846 - val_loss: 0.2804 - val_accuracy: 0.8778
 - val_kappa:  0.5905

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 22/30
 - 2s - loss: 0.2786 - accuracy: 0.8818 - val_loss: 0.2641 - val_accuracy: 0.8902
 - val_kappa:  0.6263

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 23/30
 - 2s - loss: 0.2681 - accuracy: 0.8871 - val_loss: 0.2980 - val_accuracy: 0.8625
 - val_kappa:  0.4355

Validation kappa did not improved from 0.6263218910461219
Epoch 24/30
 - 2s - loss: 0.2724 - accuracy: 0.8857 - val_loss: 0.3226 - val_accuracy: 0.8484
 - val_kappa:  0.3480

Validation kappa did not improved from 0.6263218910461219
Epoch 25/30
 - 3s - loss: 0.2618 - accuracy: 0.8902 - val_loss: 0.2681 - val_accuracy: 0.8942
 - val_kappa:  0.6039

Validation kappa did not improved from 0.6263218910461219
Epoch 26/30
 - 3s - loss: 0.2568 - accuracy: 0.8943 - val_loss: 0.2554 - val_accuracy: 0.8956
 - val_kappa:  0.6749

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
Epoch 27/30
 - 2s - loss: 0.2586 - accuracy: 0.8917 - val_loss: 0.2538 - val_accuracy: 0.9011
 - val_kappa:  0.6477

Validation kappa did not improved from 0.674917634283881
Epoch 28/30
 - 2s - loss: 0.2533 - accuracy: 0.8934 - val_loss: 0.2584 - val_accuracy: 0.8989
 - val_kappa:  0.6324

Validation kappa did not improved from 0.674917634283881
Epoch 29/30
 - 3s - loss: 0.2507 - accuracy: 0.8969 - val_loss: 0.2625 - val_accuracy: 0.8967
 - val_kappa:  0.6185

Validation kappa did not improved from 0.674917634283881
Epoch 30/30
 - 2s - loss: 0.2467 - accuracy: 0.8976 - val_loss: 0.2453 - val_accuracy: 0.9018
 - val_kappa:  0.6931

Validation Kappa has improved. Saving model to /content/drive/My Drive/models/baseline1.h5...
```

In [ ]:
```
baseline = baseline_model()
baseline.load_weights("/content/drive/My Drive/models/baseline1.h5")
result1 = baseline.evaluate(x_validation,labels_validation)
y_pred = baseline.predict(x_validation)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result1[0],4),np.round(result1[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

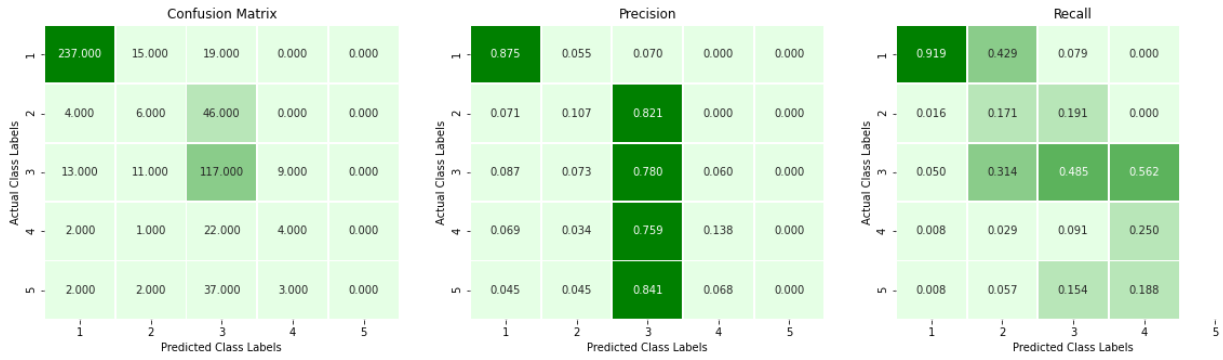
```
550/550 [==============================] - 6s 12ms/step
After running the model for 30 epochs we got loss = 0.2453 Accuracy = 0.9018 kappa_score = 0.6931 on validation data
```

In [ ]:
```
ytrain_baseline = baseline.predict(x_train)
ytrain_baseline = test_prediction(ytrain_baseline)
print("First five data points predictions in training:",ytrain_baseline[:5])
print("length of traindata prediction:",ytrain_baseline.shape,"\n")

yvalidation_baseline = baseline.predict(x_validation)
yvalidation_baseline = test_prediction(yvalidation_baseline)
print("First five data points predictions in validation:",yvalidation_baseline[:5])
print("length of validation data prediction:",yvalidation_baseline.shape,"\n")

ytest_baseline = baseline.predict(x_test)
ytest_baseline = test_prediction(ytest_baseline)
print("First five data points predictions in test:",ytest_baseline[:5])
print("length of test data prediction:",ytest_baseline.shape)
```

```
First five data points predictions in training: [0 2 0 2 2]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 2 2 2]
length of validation data prediction: (550,)

First five data points predictions in test: [1 3 2 2 0]
length of test data prediction: (1928,)
```

In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```

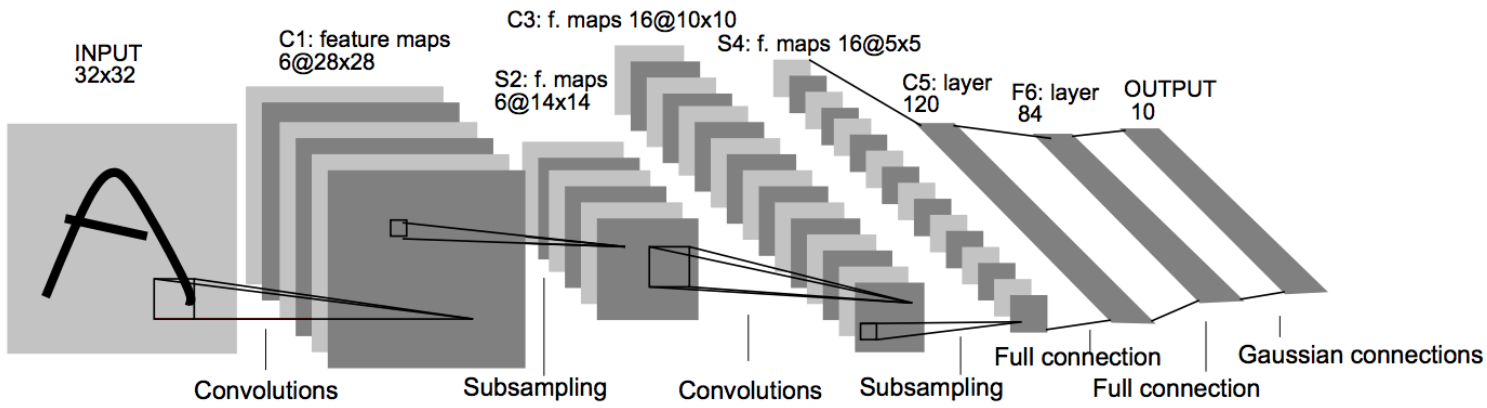
```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:26: RuntimeWarning: invalid value encountered in true_divide
```



In [ ]:
```
%load_ext tensorboard
%tensorboard --logdir="/content/drive/My Drive/baseline1"
```

Tensorboard Visualization

AlexNet Architecture:



Highlights of Alexnet:

1. Using Relu instead of Tanh to add non-linearity.
2. Local Response Normalization.
3. Overlapping Pooling.
4. Dropouts.
5. Training on Multiple GPU's
   i) Model Parallelism
   ii) Data Parallelism

In [ ]:
```
def AlexNet():
    ''' This function is used for building an alexnet architecture '''
    model = Sequential(name='alexnet')
    model.add(Conv2D(96,(11,11),strides = (4,4), padding = 'valid', activation = 'relu', kernel_initializer = 'he_normal',input_shape = (512,512,3)))
    model.add(MaxPooling2D(pool_size = (3,3), strides = (2,2)))
    model.add(Conv2D(256, kernel_size = (5,5), padding = 'same', activation = 'relu'))
    model.add(MaxPooling2D(pool_size = (3,3), strides = (2,2)))
    model.add(Conv2D(384, kernel_size = (3,3), padding = 'same', activation= 'relu'))
    model.add(Conv2D(384, kernel_size = (3,3), padding = 'same', activation = 'relu'))
    model.add(Conv2D(256, kernel_size = (3,3), padding = 'same', activation = 'relu'))
    model.add(MaxPooling2D(pool_size = (3,3), strides = (2,2)))
    model.add(Flatten())
    model.add(Dense(9216, activation = 'relu'))
    model.add(Dropout(0.5))
    model.add(Dense(4096, activation = 'relu'))
    model.add(Dropout(0.5))
    model.add(Dense(2048, activation = 'relu'))
    model.add(Dense(5, activation = 'sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00001), metrics=['accuracy'])
    return model
```

In [ ]:
```
alex_net = AlexNet()
alex_net.summary()
```

```
Model: "alexnet"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_12 (Conv2D)           (None, 126, 126, 96)      34944
_________________________________________________________________
max_pooling2d_9 (MaxPooling2 (None, 62, 62, 96)        0
_________________________________________________________________
conv2d_13 (Conv2D)           (None, 62, 62, 256)       614656
_________________________________________________________________
max_pooling2d_10 (MaxPooling (None, 30, 30, 256)       0
_________________________________________________________________
conv2d_14 (Conv2D)           (None, 30, 30, 384)       885120
_________________________________________________________________
conv2d_15 (Conv2D)           (None, 30, 30, 384)       1327488
_________________________________________________________________
conv2d_16 (Conv2D)           (None, 30, 30, 256)       884992
_________________________________________________________________
max_pooling2d_11 (MaxPooling (None, 14, 14, 256)       0
_________________________________________________________________
flatten_1 (Flatten)          (None, 50176)             0
_________________________________________________________________
dense_12 (Dense)             (None, 9216)              462431232
_________________________________________________________________
dropout_4 (Dropout)          (None, 9216)              0
_________________________________________________________________
dense_13 (Dense)             (None, 4096)              37752832
_________________________________________________________________
dropout_5 (Dropout)          (None, 4096)              0
_________________________________________________________________
dense_14 (Dense)             (None, 2048)              8390656
_________________________________________________________________
dense_15 (Dense)             (None, 5)                 10245
=================================================================
Total params: 512,332,165
Trainable params: 512,332,165
Non-trainable params: 0
_________________________________________________________________
```

In [ ]:
```
kappa_metrics = Metrics('/content/drive/My Drive/models/alexnet.h5')
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/alexnet')
call_backs = [kappa_metrics, tensorboard]
history = alex_net.fit(x_train, labels_train, epochs = 30, batch_size = 8, verbose = 2, validation_data = (x_validation, labels_validation), callbacks = call_backs)
```

```
Train on 3112 samples, validate on 550 samples
Epoch 1/30
 - 72s - loss: 2.1060 - accuracy: 0.7490 - val_loss: 0.4604 - val_accuracy: 0.8385
 - val_kappa:  0.3017

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 2/30
 - 71s - loss: 0.4699 - accuracy: 0.8148 - val_loss: 0.2935 - val_accuracy: 0.8949
 - val_kappa:  0.6671

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 3/30
 - 71s - loss: 0.3258 - accuracy: 0.8694 - val_loss: 0.2647 - val_accuracy: 0.8909
 - val_kappa:  0.5735

                Validation kappa did not improved from 0.6671219716063006
Epoch 4/30
 - 71s - loss: 0.2843 - accuracy: 0.8869 - val_loss: 0.2371 - val_accuracy: 0.9051
 - val_kappa:  0.6867

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 5/30
 - 71s - loss: 0.2610 - accuracy: 0.8973 - val_loss: 0.2348 - val_accuracy: 0.9076
 - val_kappa:  0.7236

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 6/30
 - 71s - loss: 0.2593 - accuracy: 0.8950 - val_loss: 0.2345 - val_accuracy: 0.9105
 - val_kappa:  0.7111

                Validation kappa did not improved from 0.7235767574021363
Epoch 7/30
 - 71s - loss: 0.2427 - accuracy: 0.9044 - val_loss: 0.2546 - val_accuracy: 0.9011
 - val_kappa:  0.7140

                Validation kappa did not improved from 0.7235767574021363
Epoch 8/30
 - 71s - loss: 0.2355 - accuracy: 0.9076 - val_loss: 0.2248 - val_accuracy: 0.9113
 - val_kappa:  0.7146

                Validation kappa did not improved from 0.7235767574021363
Epoch 9/30
 - 71s - loss: 0.2295 - accuracy: 0.9091 - val_loss: 0.2257 - val_accuracy: 0.9098
 - val_kappa:  0.7061

                Validation kappa did not improved from 0.7235767574021363
Epoch 10/30
 - 71s - loss: 0.2265 - accuracy: 0.9100 - val_loss: 0.2164 - val_accuracy: 0.9171
 - val_kappa:  0.7400

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 11/30
 - 71s - loss: 0.2167 - accuracy: 0.9152 - val_loss: 0.2213 - val_accuracy: 0.9109
 - val_kappa:  0.7438

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 12/30
 - 71s - loss: 0.2145 - accuracy: 0.9137 - val_loss: 0.2200 - val_accuracy: 0.9142
 - val_kappa:  0.7254

                Validation kappa did not improved from 0.743836149465269
Epoch 13/30
 - 71s - loss: 0.2149 - accuracy: 0.9109 - val_loss: 0.2140 - val_accuracy: 0.9153
 - val_kappa:  0.7327

                Validation kappa did not improved from 0.743836149465269
Epoch 14/30
 - 71s - loss: 0.2055 - accuracy: 0.9177 - val_loss: 0.2087 - val_accuracy: 0.9164
 - val_kappa:  0.7300

                Validation kappa did not improved from 0.743836149465269
Epoch 15/30
 - 71s - loss: 0.1963 - accuracy: 0.9204 - val_loss: 0.2222 - val_accuracy: 0.9105
 - val_kappa:  0.7580

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 16/30
 - 71s - loss: 0.1968 - accuracy: 0.9196 - val_loss: 0.2172 - val_accuracy: 0.9178
 - val_kappa:  0.7620

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 17/30
 - 71s - loss: 0.1856 - accuracy: 0.9242 - val_loss: 0.2131 - val_accuracy: 0.9153
 - val_kappa:  0.7519

                Validation kappa did not improved from 0.7619717104539285
Epoch 18/30
 - 71s - loss: 0.1837 - accuracy: 0.9231 - val_loss: 0.2031 - val_accuracy: 0.9160
 - val_kappa:  0.7144

                Validation kappa did not improved from 0.7619717104539285
Epoch 19/30
 - 71s - loss: 0.1807 - accuracy: 0.9238 - val_loss: 0.2307 - val_accuracy: 0.9065
 - val_kappa:  0.7572

                Validation kappa did not improved from 0.7619717104539285
Epoch 20/30
 - 71s - loss: 0.1747 - accuracy: 0.9261 - val_loss: 0.2496 - val_accuracy: 0.9040
 - val_kappa:  0.6351

                Validation kappa did not improved from 0.7619717104539285
Epoch 21/30
 - 71s - loss: 0.1702 - accuracy: 0.9286 - val_loss: 0.2024 - val_accuracy: 0.9164
 - val_kappa:  0.7287

                Validation kappa did not improved from 0.7619717104539285
Epoch 22/30
 - 71s - loss: 0.1618 - accuracy: 0.9309 - val_loss: 0.2158 - val_accuracy: 0.9145
 - val_kappa:  0.7097

                Validation kappa did not improved from 0.7619717104539285
Epoch 23/30
 - 71s - loss: 0.1575 - accuracy: 0.9329 - val_loss: 0.2160 - val_accuracy: 0.9171
 - val_kappa:  0.7274

                Validation kappa did not improved from 0.7619717104539285
Epoch 24/30
 - 71s - loss: 0.1461 - accuracy: 0.9370 - val_loss: 0.2289 - val_accuracy: 0.9062
 - val_kappa:  0.6939

                Validation kappa did not improved from 0.7619717104539285
Epoch 25/30
 - 71s - loss: 0.1430 - accuracy: 0.9385 - val_loss: 0.2098 - val_accuracy: 0.9204
 - val_kappa:  0.7599

                Validation kappa did not improved from 0.7619717104539285
Epoch 26/30
 - 71s - loss: 0.1321 - accuracy: 0.9424 - val_loss: 0.2293 - val_accuracy: 0.8924
 - val_kappa:  0.7469

                Validation kappa did not improved from 0.7619717104539285
Epoch 27/30
 - 71s - loss: 0.1268 - accuracy: 0.9449 - val_loss: 0.2232 - val_accuracy: 0.9164
 - val_kappa:  0.7511

                Validation kappa did not improved from 0.7619717104539285
Epoch 28/30
 - 71s - loss: 0.1197 - accuracy: 0.9474 - val_loss: 0.2228 - val_accuracy: 0.9204
 - val_kappa:  0.7758

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/alexnet.h5...
Epoch 29/30
 - 71s - loss: 0.1142 - accuracy: 0.9529 - val_loss: 0.2162 - val_accuracy: 0.9131
 - val_kappa:  0.7735

                Validation kappa did not improved from 0.7758198246747761
Epoch 30/30
 - 71s - loss: 0.0997 - accuracy: 0.9580 - val_loss: 0.2360 - val_accuracy: 0.9185
 - val_kappa:  0.7750

                Validation kappa did not improved from 0.7758198246747761
```

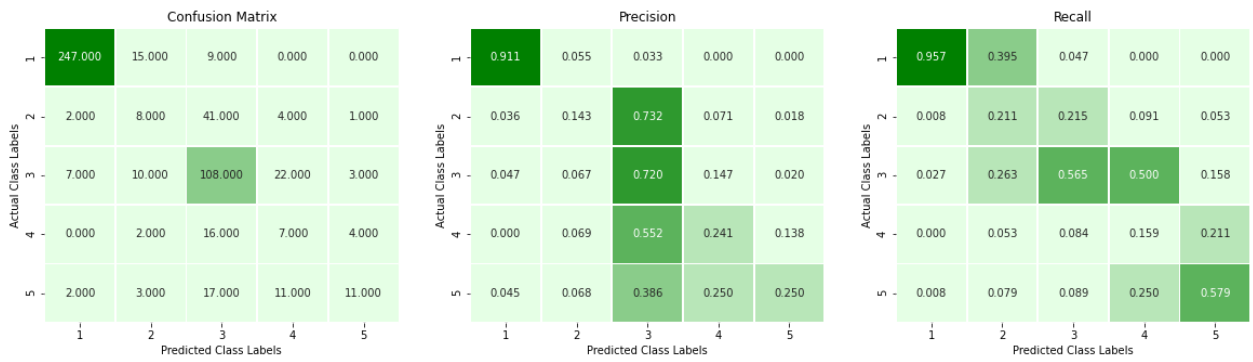
In [ ]:
```
alex_net = AlexNet()
alex_net.load_weights("/content/drive/My Drive/models/alexnet.h5")
result = alex_net.evaluate(x_validation,labels_validation)
y_pred = alex_net.predict(x_validation)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result[0],4),np.round(result[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
550/550 [==============================] - 6s 12ms/step
After running the model for 30 epochs we got loss = 0.2228 Accuracy = 0.9204 kappa_score = 0.7758 on validation data
```

In [ ]:
```
ytrain_alexnet = alex_net.predict(x_train)
ytrain_alexnet = test_prediction(ytrain_alexnet)
print("First five data points predictions in training:",ytrain_alexnet[45:50])
print("length of traindata prediction:",ytrain_alexnet.shape,"\n")

yvalidation_alexnet = alex_net.predict(x_validation)
yvalidation_alexnet = test_prediction(yvalidation_alexnet)
print("First five data points predictions in validation:",yvalidation_alexnet[45:50])
print("length of validation data prediction:",yvalidation_alexnet.shape,"\n")

ytest_alexnet = alex_net.predict(x_test)
ytest_alexnet = test_prediction(ytest_alexnet)
print("First five data points predictions in test:",ytest_alexnet[45:50])
print("length of test data prediction:",ytest_alexnet.shape)
```

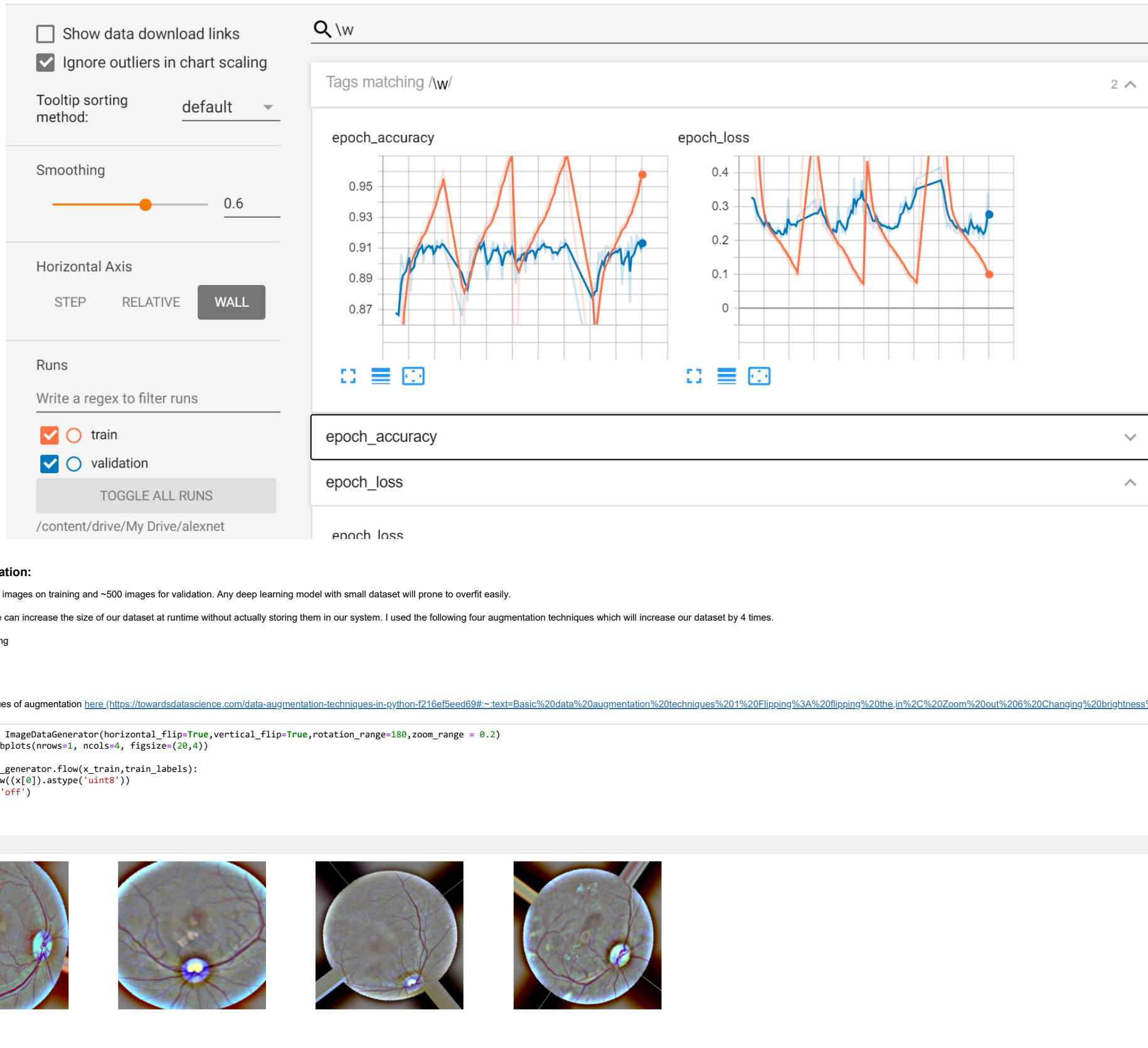
```
First five data points predictions in training: [3 3 3 3 3]
length of traindata prediction: (3112,)

First five data points predictions in validation: [3 3 3 3 3]
length of validation data prediction: (550,)

First five data points predictions in test: [3 3 3 3 3]
length of test data prediction: (1928,)
```

In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



In [ ]:
```
%load_ext tensorboard
%tensorboard --logdir="/content/drive/My Drive/alexnet"
```

Tensorboard Visualization:



Data Augmentation:

We hardly have ~3k images on training and ~500 images for validation. Any deep learning model with small dataset will prone to overfit easily.

By augmentation we can increase the size of our dataset at runtime without actually storing them in our system. I used the following four augmentation techniques which will increase our dataset by 4 times.

1. horizontal flipping
2. Vertical flipping
3. rotation_range
4. zoom range

Read more techniques of augmentation here (https://towardsdatascience.com/data-augmentation-techniques-in-python-f216ef5eed69#:~:text=Basic%20data%20augmentation%20techniques%201%20Flipping%3A%20flipping%20the,in%2C%20Zoom%20out%206%20Changing%20brightness%20or%20contrast).

In [ ]:
```
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
fig, ax = plt.subplots(nrows=1, ncols=4, figsize=(20,4))
it = 0
for x, y in data_generator.flow(x_train,train_labels):
    ax[it].imshow((x[0]).astype('uint8'))
    ax[it].axis('off')
    it += 1
    if it == 4:
        break
```

In [ ]:
```
def GAP2D():
    '''Global average pooling layer'''
    global_average_pooling = GlobalAveragePooling2D()
    return global_average_pooling
def dropout(value = 0.5):
    '''Dropout layer'''
    dropout_layer = Dropout(value)
    return dropout_layer
def dense():
    '''Dense layer'''
    dense_layer = Dense(5, activation='sigmoid')
    return dense_layer
```

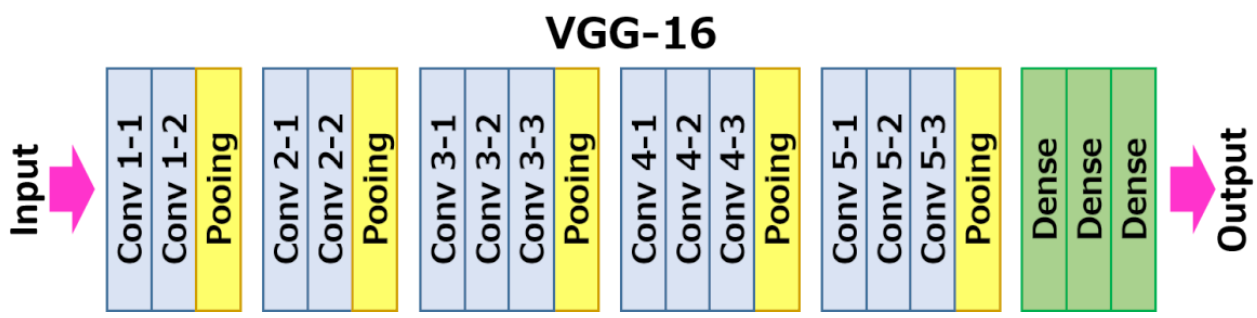
All the pretrained model architectures are extended with following three layers.

1. **GlobalAveragePooling2D:**
   From the last convolutional layer of any pretained model which generates as many feature maps as the number of target classes, and applies global average pooling to each in order to convert each feature map into one value by taking average of it.
2. **Dropout**
   Dropout works by randomly setting the outgoing edges of hidden units to 0 at each update of the training phase. Which will be used as a sort of regularization in order to avoid overfitting during model training.
3. **Dense**
   This layer contains 5 units each represents the individual class with sigmoid activation.
   Optimizer: Adam
   loss: binary cross entropy
   metric: accuracy and weighted kappa

Visual Geometry Group(VGG).

- VGG has 16 or 19 layers starting with simple 3x3 filters in first convolutional layer.
- Number of channels increased with term of 2 and Nh, Nw decreased with factor of 2.
- Though it contains lots of parameters, the model is simple with fixed 3x3 filters and stride=1 and same padding in convolutional layers.
- 2x2 filters with stride=2 in max pooling layers.

VGG-16 architecture:



In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def VGG16_(top_6 = False):
    '''This function is used for building a model architecture of pretrained vgg16 on imagenet data set.'''
    vgg = VGG16(weights = 'imagenet', include_top = False, input_shape = (512,512,3))
    if not top_6:
        for layer in vgg.layers:
            layer.trainable = False
    else:
         for layer in vgg.layers[:13]:
            layer.trainable=False
    x = global_average_pooling_layer(vgg.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(vgg.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
vgg16 = VGG16_(True)
vgg16.summary()
```

```
Model: "Vgg-16"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 16, 16, 512)       14714688
_________________________________________________________________
global_average_pooling2d_4 ( (None, 512)               0
_________________________________________________________________
dropout_8 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_18 (Dense)             (None, 5)                 2565
=================================================================
Total params: 14,717,253
Trainable params: 9,441,797
Non-trainable params: 5,275,456
_________________________________________________________________
```

Note:

```
def VGG16_(top_6 = False):
    '''
    this function is used to modify Vgg16 architecture which is trained on imagenet.
    '''
    vgg = VGG16(weights = 'imagenet',include_top = False,input_shape = (512,512,3))
    if not top_6:
        for layer in vgg.layers:
            layer.trainable = False
    else:
        for layer in vgg.layers[:13]:
            layer.trainable = False
    model = Sequential(name='VGG-16')
    model.add(vgg)
    model.add(GlobalAveragePooling2D())
    model.add(Dropout(0.5))
    model.add(Dense(5, activation='sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

I used this architecture for printing summary which is actually different from the above in order to discard pretrained layers in summary section. But the actual results will be of same on using any architecture.

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/vgg16')
kappa_metrics = Metrics('/content/drive/My Drive/models/vgg16.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
vgg16 = VGG16_(top_6 = True)
history = vgg16.fit_generator(
                   data_generator.flow(x_train, labels_train, batch_size=12),
                    steps_per_epoch=len(x_train) / 12,
                    epochs=30,
                    initial_epoch=0,
                    verbose=1,
                    validation_data=(x_validation, labels_validation),
                    validation_steps=len(x_validation) / 12,
                    callbacks=[kappa_metrics,tensorboard])
```

```
Epoch 1/30
260/259 [==============================] - 42s 163ms/step - loss: 0.4320 - accuracy: 0.8418 - val_loss: 0.3015 - val_accuracy: 0.8898
 - val_kappa:  0.5891

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 2/30
260/259 [==============================] - 35s 134ms/step - loss: 0.2500 - accuracy: 0.9017 - val_loss: 0.2467 - val_accuracy: 0.9149
 - val_kappa:  0.7218

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 3/30
260/259 [==============================] - 35s 136ms/step - loss: 0.2234 - accuracy: 0.9116 - val_loss: 0.1838 - val_accuracy: 0.9335
 - val_kappa:  0.8043

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 4/30
260/259 [==============================] - 35s 134ms/step - loss: 0.1997 - accuracy: 0.9200 - val_loss: 0.2105 - val_accuracy: 0.9229
 - val_kappa:  0.7506

                Validation kappa did not improved from 0.8042936009740989
Epoch 5/30
260/259 [==============================] - 35s 134ms/step - loss: 0.1946 - accuracy: 0.9255 - val_loss: 0.2076 - val_accuracy: 0.9244
 - val_kappa:  0.7580

                Validation kappa did not improved from 0.8042936009740989
Epoch 6/30
260/259 [==============================] - 36s 137ms/step - loss: 0.1769 - accuracy: 0.9299 - val_loss: 0.1691 - val_accuracy: 0.9338
 - val_kappa:  0.8083

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 7/30
260/259 [==============================] - 35s 136ms/step - loss: 0.1732 - accuracy: 0.9331 - val_loss: 0.1627 - val_accuracy: 0.9371
 - val_kappa:  0.8318

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 8/30
260/259 [==============================] - 35s 135ms/step - loss: 0.1623 - accuracy: 0.9368 - val_loss: 0.1994 - val_accuracy: 0.9302
 - val_kappa:  0.7897

                Validation kappa did not improved from 0.8317894163970723
Epoch 9/30
260/259 [==============================] - 35s 135ms/step - loss: 0.1672 - accuracy: 0.9349 - val_loss: 0.1472 - val_accuracy: 0.9411
 - val_kappa:  0.8367

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 10/30
260/259 [==============================] - 35s 136ms/step - loss: 0.1573 - accuracy: 0.9375 - val_loss: 0.1451 - val_accuracy: 0.9491
 - val_kappa:  0.8708

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 11/30
260/259 [==============================] - 35s 135ms/step - loss: 0.1500 - accuracy: 0.9406 - val_loss: 0.1848 - val_accuracy: 0.9393
 - val_kappa:  0.8291

                Validation kappa did not improved from 0.8707919563876197
Epoch 12/30
260/259 [==============================] - 35s 136ms/step - loss: 0.1419 - accuracy: 0.9441 - val_loss: 0.2015 - val_accuracy: 0.9320
 - val_kappa:  0.8064

                Validation kappa did not improved from 0.8707919563876197
Epoch 13/30
260/259 [==============================] - 35s 135ms/step - loss: 0.1454 - accuracy: 0.9438 - val_loss: 0.1622 - val_accuracy: 0.9418
 - val_kappa:  0.8539

                Validation kappa did not improved from 0.8707919563876197
Epoch 14/30
260/259 [==============================] - 35s 134ms/step - loss: 0.1431 - accuracy: 0.9443 - val_loss: 0.1374 - val_accuracy: 0.9502
 - val_kappa:  0.8834

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/vgg16.h5...
Epoch 15/30
260/259 [==============================] - 35s 134ms/step - loss: 0.1410 - accuracy: 0.9438 - val_loss: 0.1528 - val_accuracy: 0.9462
 - val_kappa:  0.8572

                Validation kappa did not improved from 0.8834187233045386
Epoch 16/30
260/259 [==============================] - 35s 134ms/step - loss: 0.1359 - accuracy: 0.9453 - val_loss: 0.1629 - val_accuracy: 0.9400
 - val_kappa:  0.8226

                Validation kappa did not improved from 0.8834187233045386
Epoch 17/30
260/259 [==============================] - 35s 133ms/step - loss: 0.1311 - accuracy: 0.9476 - val_loss: 0.1399 - val_accuracy: 0.9484
 - val_kappa:  0.8672

                Validation kappa did not improved from 0.8834187233045386
Epoch 18/30
260/259 [==============================] - 35s 134ms/step - loss: 0.1294 - accuracy: 0.9480 - val_loss: 0.1464 - val_accuracy: 0.9473
 - val_kappa:  0.8556

                Validation kappa did not improved from 0.8834187233045386
Epoch 19/30
161/259 [=================>............] - ETA: 12s - loss: 0.1299 - accuracy: 0.9485
```

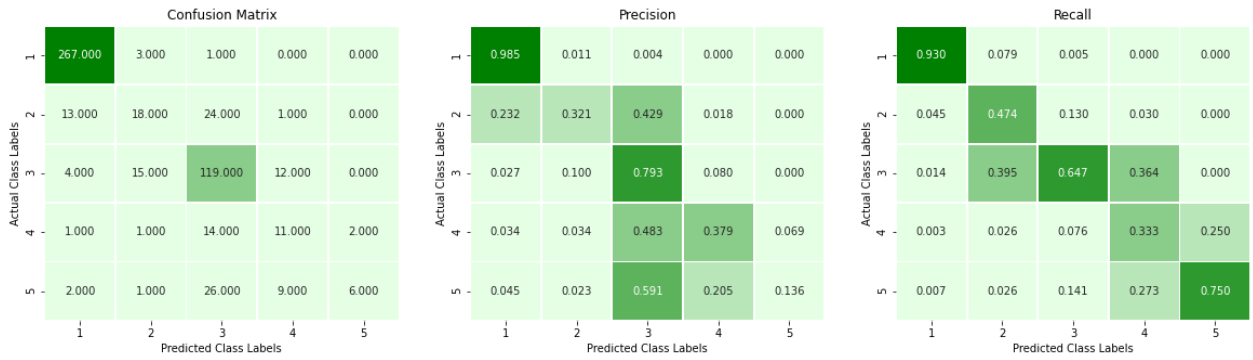
In [ ]:
```
vgg16 = VGG16_(top_6 = True)
vgg16.load_weights("/content/drive/My Drive/models/vgg16.h5")
result = vgg16.evaluate(x_validation,labels_validation)
y_pred = vgg16.predict(x_validation)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result[0],4),np.round(result[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5 (https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5)
58892288/58889256 [==============================] - 2s 0us/step
550/550 [==============================] - 9s 16ms/step
After running the model for 30 epochs we got loss = 0.1409 Accuracy = 0.9535 kappa_score = 0.8931 on validation data
```

In [ ]:
```
ytrain_vgg16 = vgg16.predict(x_train)
ytrain_vgg16 = test_prediction(ytrain_vgg16)
print("First five data points predictions in training:",ytrain_vgg16[:5])
print("length of traindata prediction:",ytrain_vgg16.shape,"\n")

yvalidation_vgg16 = vgg16.predict(x_validation)
yvalidation_vgg16 = test_prediction(yvalidation_vgg16)
print("First five data points predictions in validation:",yvalidation_vgg16[:5])
print("length of validation data prediction:",yvalidation_vgg16.shape,"\n")

ytest_vgg16 = vgg16.predict(x_test)
ytest_vgg16 = test_prediction(ytest_vgg16)
print("First five data points predictions in test:",ytest_vgg16[:5])
print("length of test data prediction:",ytest_vgg16.shape)
```

```
First five data points predictions in training: [0 2 0 2 2]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 0 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 2 3 2 2]
length of test data prediction: (1928,)
```

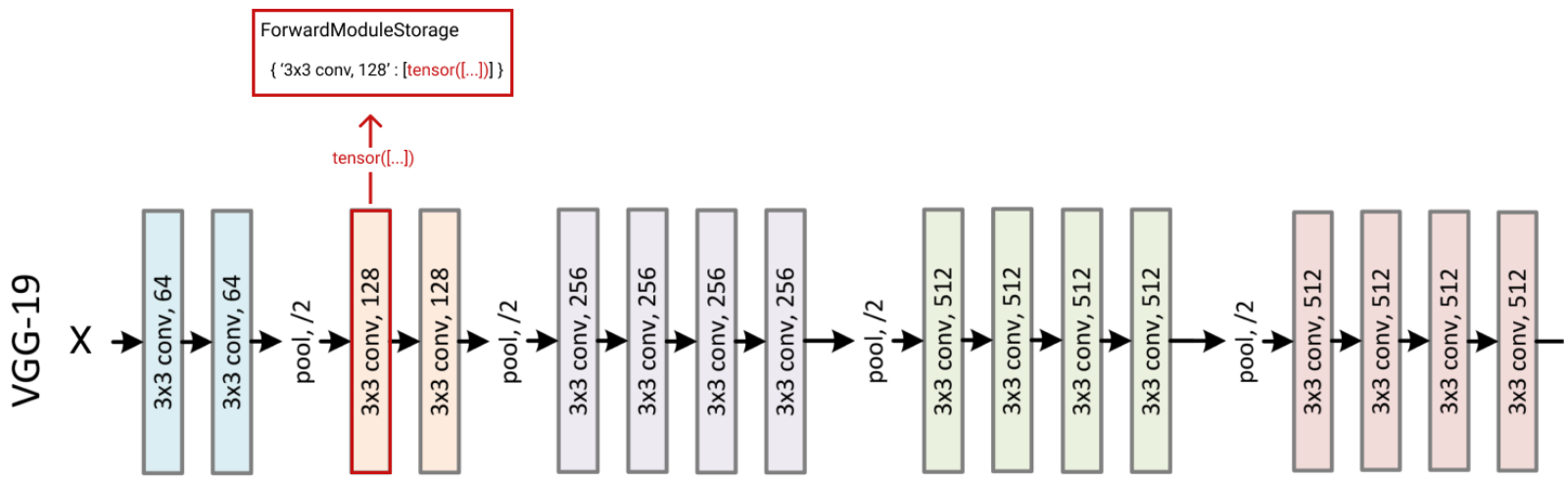
In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/vgg16'
```

Tensorboard Visualization:

VGG-19 Architecture:



In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def VGG19_(top_6 = False):
    '''This function is used for building a model architecture of pretrained Vgg19 on imagenet data set.'''
    vgg = VGG19(weights = 'imagenet', include_top = False, input_shape = (512,512,3))
    if not top_6:
        for layer in vgg.layers:
            layer.trainable = False
    else:
        for layer in vgg.layers[:13]:
            layer.trainable=False
    x = global_average_pooling_layer(vgg.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(vgg.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
vgg19 = VGG19_(top_6 = True)
vgg19.summary()
```

```
Model: "Vgg-19"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg19 (Functional)           (None, 16, 16, 512)       20024384
_________________________________________________________________
global_average_pooling2d_6 ( (None, 512)               0
_________________________________________________________________
dropout_10 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_20 (Dense)             (None, 5)                 2565
=================================================================
Total params: 20,026,949
Trainable params: 16,521,221
Non-trainable params: 3,505,728
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/models/vgg19')
kappa_metrics = Metrics('/content/drive/My Drive/vgg19.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
vgg19 = VGG19_(top_6 = True)
history = vgg19.fit_generator(
                  data_generator.flow(x_train, labels_train, batch_size=8),
                   steps_per_epoch=len(x_train) / 8,
                   epochs=30,
                   initial_epoch=0,
                   verbose=1,
                   validation_data=(x_validation, labels_validation),
                   validation_steps=len(x_validation) / 8,
                   callbacks=[kappa_metrics,tensorboard], class_weight = class_weights)
```

```
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5 (https://github.com/fchollet/deep-learning-models/releases/download/v0.1/vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5)
80142336/80134624 [==============================] - 3s 0us/step
Epoch 1/30
  5/389 [..............................] - ETA: 1:24 - loss: 3.5918 - accuracy: 0.5250

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (0.124578). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 41s 105ms/step - loss: 0.4839 - accuracy: 0.8208 - val_loss: 0.2136 - val_accuracy: 0.9167
 - val_kappa:  0.8014

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 2/30
389/389 [==============================] - 40s 102ms/step - loss: 0.2522 - accuracy: 0.9001 - val_loss: 0.1923 - val_accuracy: 0.9215
 - val_kappa:  0.8168

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 3/30
389/389 [==============================] - 40s 102ms/step - loss: 0.2256 - accuracy: 0.9104 - val_loss: 0.1720 - val_accuracy: 0.9353
 - val_kappa:  0.8326

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 4/30
389/389 [==============================] - 40s 102ms/step - loss: 0.2022 - accuracy: 0.9216 - val_loss: 0.1540 - val_accuracy: 0.9382
 - val_kappa:  0.8487

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 5/30
389/389 [==============================] - 40s 102ms/step - loss: 0.1925 - accuracy: 0.9274 - val_loss: 0.1638 - val_accuracy: 0.9335
 - val_kappa:  0.8348

                Validation kappa did not improved from 0.8487103779800396
Epoch 6/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1790 - accuracy: 0.9292 - val_loss: 0.1518 - val_accuracy: 0.9400
 - val_kappa:  0.8310

                Validation kappa did not improved from 0.8487103779800396
Epoch 7/30
389/389 [==============================] - 39s 100ms/step - loss: 0.1765 - accuracy: 0.9328 - val_loss: 0.1407 - val_accuracy: 0.9415
 - val_kappa:  0.8474

                Validation kappa did not improved from 0.8487103779800396
Epoch 8/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1720 - accuracy: 0.9298 - val_loss: 0.1403 - val_accuracy: 0.9473
 - val_kappa:  0.8785

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 9/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1648 - accuracy: 0.9349 - val_loss: 0.1383 - val_accuracy: 0.9484
 - val_kappa:  0.8840

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 10/30
389/389 [==============================] - 40s 102ms/step - loss: 0.1584 - accuracy: 0.9361 - val_loss: 0.1530 - val_accuracy: 0.9440
 - val_kappa:  0.8467

                Validation kappa did not improved from 0.8839713852339875
Epoch 11/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1585 - accuracy: 0.9376 - val_loss: 0.1489 - val_accuracy: 0.9429
 - val_kappa:  0.8469

                Validation kappa did not improved from 0.8839713852339875
Epoch 12/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1526 - accuracy: 0.9395 - val_loss: 0.1399 - val_accuracy: 0.9480
 - val_kappa:  0.8698

                Validation kappa did not improved from 0.8839713852339875
Epoch 13/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1490 - accuracy: 0.9397 - val_loss: 0.1283 - val_accuracy: 0.9513
 - val_kappa:  0.8732

                Validation kappa did not improved from 0.8839713852339875
Epoch 14/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1497 - accuracy: 0.9413 - val_loss: 0.1388 - val_accuracy: 0.9469
 - val_kappa:  0.8737

                Validation kappa did not improved from 0.8839713852339875
Epoch 15/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1426 - accuracy: 0.9437 - val_loss: 0.1263 - val_accuracy: 0.9531
 - val_kappa:  0.8790

                Validation kappa did not improved from 0.8839713852339875
Epoch 16/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1462 - accuracy: 0.9431 - val_loss: 0.1249 - val_accuracy: 0.9531
 - val_kappa:  0.8927

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 17/30
389/389 [==============================] - 40s 102ms/step - loss: 0.1367 - accuracy: 0.9458 - val_loss: 0.1480 - val_accuracy: 0.9465
 - val_kappa:  0.8552

                Validation kappa did not improved from 0.8927114736655435
Epoch 18/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1391 - accuracy: 0.9445 - val_loss: 0.1241 - val_accuracy: 0.9538
 - val_kappa:  0.8850

                Validation kappa did not improved from 0.8927114736655435
Epoch 19/30
389/389 [==============================] - 39s 100ms/step - loss: 0.1375 - accuracy: 0.9469 - val_loss: 0.1623 - val_accuracy: 0.9447
 - val_kappa:  0.8497

                Validation kappa did not improved from 0.8927114736655435
Epoch 20/30
389/389 [==============================] - 39s 100ms/step - loss: 0.1360 - accuracy: 0.9469 - val_loss: 0.1351 - val_accuracy: 0.9502
 - val_kappa:  0.8610

                Validation kappa did not improved from 0.8927114736655435
Epoch 21/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1353 - accuracy: 0.9472 - val_loss: 0.1331 - val_accuracy: 0.9469
 - val_kappa:  0.8660

                Validation kappa did not improved from 0.8927114736655435
Epoch 22/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1284 - accuracy: 0.9497 - val_loss: 0.1200 - val_accuracy: 0.9578
 - val_kappa:  0.9010

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 23/30
389/389 [==============================] - 40s 102ms/step - loss: 0.1297 - accuracy: 0.9505 - val_loss: 0.1309 - val_accuracy: 0.9498
 - val_kappa:  0.8725

                Validation kappa did not improved from 0.9010188984847312
Epoch 24/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1233 - accuracy: 0.9510 - val_loss: 0.1295 - val_accuracy: 0.9487
 - val_kappa:  0.9052

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 25/30
389/389 [==============================] - 40s 102ms/step - loss: 0.1192 - accuracy: 0.9525 - val_loss: 0.1102 - val_accuracy: 0.9575
 - val_kappa:  0.9128

                Validation Kappa has improved. Saving model to /content/drive/My Drive/vgg19.h5...
Epoch 26/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1271 - accuracy: 0.9487 - val_loss: 0.1296 - val_accuracy: 0.9531
 - val_kappa:  0.8928

                Validation kappa did not improved from 0.9127961096324241
Epoch 27/30
389/389 [==============================] - 39s 100ms/step - loss: 0.1208 - accuracy: 0.9534 - val_loss: 0.1293 - val_accuracy: 0.9567
 - val_kappa:  0.9020

                Validation kappa did not improved from 0.9127961096324241
Epoch 28/30
389/389 [==============================] - 39s 100ms/step - loss: 0.1187 - accuracy: 0.9544 - val_loss: 0.1306 - val_accuracy: 0.9513
 - val_kappa:  0.9043

                Validation kappa did not improved from 0.9127961096324241
Epoch 29/30
389/389 [==============================] - 39s 101ms/step - loss: 0.1131 - accuracy: 0.9543 - val_loss: 0.1201 - val_accuracy: 0.9549
 - val_kappa:  0.9088

                Validation kappa did not improved from 0.9127961096324241
```

```
Epoch 30/30
389/389 [==============================] - 39s 100ms/step - loss: 0.1160 - accuracy: 0.9531 - val_loss: 0.1221 - val_accuracy: 0.9542
 - val_kappa:  0.9053

                Validation kappa did not improved from 0.9127961096324241
```

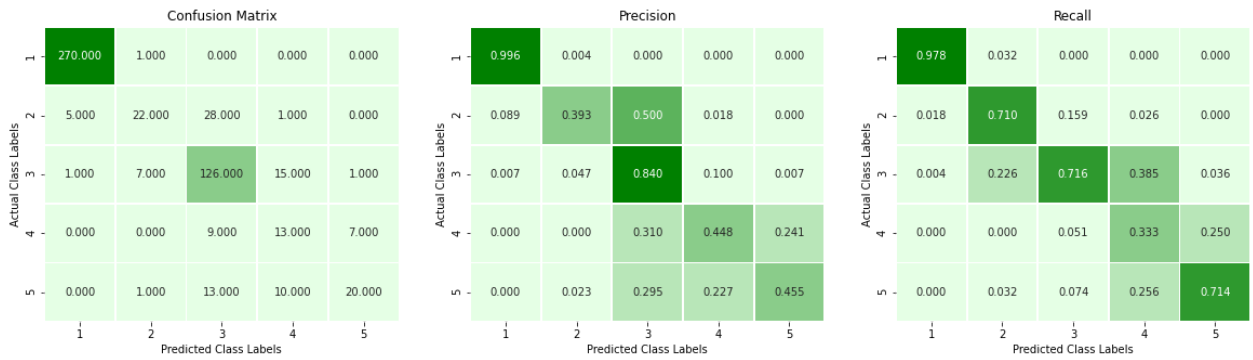
In [ ]:
```
vgg19 = VGG19_(top_6 = True)
vgg19.load_weights("/content/drive/My Drive/vgg19.h5")
result = vgg19.evaluate(x_validation,labels_validation)
y_pred = vgg19.predict(x_validation)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result[0],4),np.round(result[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
550/550 [==============================] - 6s 11ms/step
After running the model for 30 epochs we got loss = 0.1102 Accuracy = 0.9575 kappa_score = 0.9128 on validation data
```

In [ ]:
```
ytrain_vgg19 = vgg19.predict(x_train)
ytrain_vgg19 = test_prediction(ytrain_vgg19)
print("First five data points predictions in training:",ytrain_vgg19[:5])
print("length of traindata prediction:",ytrain_vgg19.shape,"\n")

yvalidation_vgg19 = vgg19.predict(x_validation)
yvalidation_vgg19 = test_prediction(yvalidation_vgg19)
print("First five data points predictions in validation:",yvalidation_vgg19[:5])
print("length of validation data prediction:",yvalidation_vgg19.shape,"\n")

ytest_vgg19 = vgg19.predict(x_test)
ytest_vgg19 = test_prediction(ytest_vgg19)
print("First five data points predictions in test:",ytest_vgg19[:5])
print("length of test data prediction:",ytest_vgg19.shape)
```

```
First five data points predictions in training: [0 2 0 2 2]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 0 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 3 2 2 2]
length of test data prediction: (1928,)
```

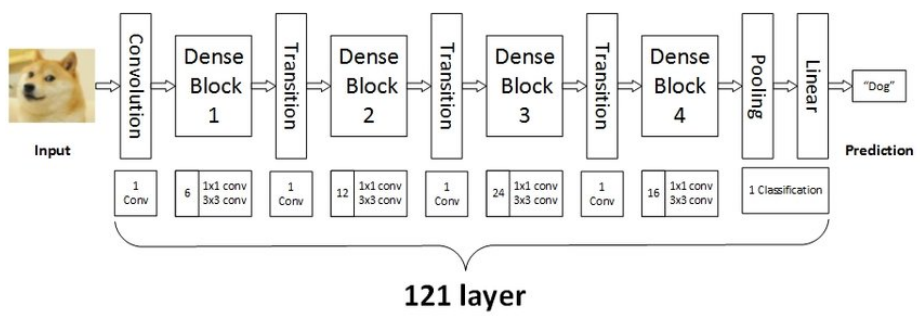
In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/vgg19'
```

Tensorboard Visualization:

DenseNet Architecture:



- Densenet was developed specifically to improve the declined accuracy caused by the vanishing gradient in in high-level neural networks.
- Due to the longer path between the input and output layer, the information vanishes before reaching its destination.
- It is a 5-layer dense block with a growth rate of k = 4.
- An output of the previous layer acts as an input of the second layer by using composite function operation. This composite operation consists of the convolution layer, pooling layer, batch normalization, and non-linear activation layer.

DenseNet with 121 layers:

=> 5+(6+12+24+16)*2 = 121

- 5 - Convolutional and Pooling layer
- 3 - Transition layers(6,12,24)
- 1 - Classification layer(16)
- 2 - DenseBlock(1x1 and 3x3 conv)

In [33]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def DenseNet():
    '''This function is used for building a model architecture of pretrained Densenet121 on imagenet data set.'''
    densenet = DenseNet121(weights='imagenet', include_top=False, input_shape=(512,512,3))
    x = global_average_pooling_layer(densenet.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(densenet.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [17]:
```
densenet = DenseNet()
densenet.summary()
```

```
Model: "DenseNet-121"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet121 (Functional)     (None, 16, 16, 1024)      7037504
_________________________________________________________________
global_average_pooling2d_1 ( (None, 1024)              0
_________________________________________________________________
dropout_1 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 5125
=================================================================
Total params: 7,042,629
Trainable params: 6,958,981
Non-trainable params: 83,648
_________________________________________________________________
```

In [ ]:
```
densenet = DenseNet()
densenet.summary()
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/densenet/densenet121_weights_tf_dim_ordering_tf_kernels_notop.h5 (https://storage.googleapis.com/tensorflow/keras-applications/densenet/densenet121_weights_tf_dim_ordering_tf_kernels_notop.h5)
29089792/29084464 [==============================] - 1s 0us/step
Model: "DenseNet-121"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
densenet121 (Functional)     (None, 16, 16, 1024)      7037504
_________________________________________________________________
global_average_pooling2d_7 ( (None, 1024)              0
_________________________________________________________________
dropout_11 (Dropout)         (None, 1024)              0
_________________________________________________________________
dense_21 (Dense)             (None, 5)                 5125
=================================================================
Total params: 7,042,629
Trainable params: 6,958,981
Non-trainable params: 83,648
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/densenet')
kappa_metrics = Metrics('/content/drive/My Drive/models/densenet.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
densenet = DenseNet()
history = densenet.fit_generator(
                    data_generator.flow(x_train, labels_train, batch_size=8),
                     steps_per_epoch=len(x_train) / 8,
                     epochs=30,
                     initial_epoch=0,
                     verbose=1,
                     validation_data=(x_validation, labels_validation),
                     validation_steps=len(x_validation) / 8,
                     callbacks=[kappa_metrics, tensorboard], class_weight = class_weights)
```

```
Epoch 1/30
  2/389 [..............................] - ETA: 1:54:05 - loss: 0.8410 - accuracy: 0.5250

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (0.644158). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 143s 367ms/step - loss: 0.3514 - accuracy: 0.8506 - val_loss: 0.1868 - val_accuracy: 0.9265
 - val_kappa:  0.7847

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/densenet.h5...
Epoch 2/30
389/389 [==============================] - 102s 263ms/step - loss: 0.2174 - accuracy: 0.9158 - val_loss: 0.1346 - val_accuracy: 0.9491
 - val_kappa:  0.8877

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/densenet.h5...
Epoch 3/30
389/389 [==============================] - 102s 262ms/step - loss: 0.1780 - accuracy: 0.9287 - val_loss: 0.1303 - val_accuracy: 0.9487
 - val_kappa:  0.8860

                Validation kappa did not improved from 0.8877281059701161
Epoch 4/30
389/389 [==============================] - 101s 260ms/step - loss: 0.1635 - accuracy: 0.9352 - val_loss: 0.1311 - val_accuracy: 0.9476
 - val_kappa:  0.8971

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/densenet.h5...
Epoch 5/30
389/389 [==============================] - 101s 260ms/step - loss: 0.1480 - accuracy: 0.9421 - val_loss: 0.1435 - val_accuracy: 0.9400
 - val_kappa:  0.8848

                Validation kappa did not improved from 0.8971187463721583
Epoch 6/30
389/389 [==============================] - 101s 260ms/step - loss: 0.1426 - accuracy: 0.9467 - val_loss: 0.1151 - val_accuracy: 0.9553
 - val_kappa:  0.8892

                Validation kappa did not improved from 0.8971187463721583
Epoch 7/30
389/389 [==============================] - 101s 260ms/step - loss: 0.1409 - accuracy: 0.9477 - val_loss: 0.1202 - val_accuracy: 0.9509
 - val_kappa:  0.8729

                Validation kappa did not improved from 0.8971187463721583
Epoch 8/30
389/389 [==============================] - 101s 259ms/step - loss: 0.1325 - accuracy: 0.9499 - val_loss: 0.1074 - val_accuracy: 0.9615
 - val_kappa:  0.9208

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/densenet.h5...
Epoch 9/30
389/389 [==============================] - 101s 261ms/step - loss: 0.1231 - accuracy: 0.9517 - val_loss: 0.1182 - val_accuracy: 0.9545
 - val_kappa:  0.8979

                Validation kappa did not improved from 0.9208342709556043
Epoch 10/30
389/389 [==============================] - 101s 261ms/step - loss: 0.1272 - accuracy: 0.9528 - val_loss: 0.1046 - val_accuracy: 0.9596
 - val_kappa:  0.9076

                Validation kappa did not improved from 0.9208342709556043
Epoch 11/30
389/389 [==============================] - 101s 261ms/step - loss: 0.1210 - accuracy: 0.9557 - val_loss: 0.1095 - val_accuracy: 0.9596
 - val_kappa:  0.9062

                Validation kappa did not improved from 0.9208342709556043
Epoch 12/30
389/389 [==============================] - 101s 260ms/step - loss: 0.1206 - accuracy: 0.9539 - val_loss: 0.1025 - val_accuracy: 0.9611
 - val_kappa:  0.9147

                Validation kappa did not improved from 0.9208342709556043
Epoch 13/30
389/389 [==============================] - 101s 260ms/step - loss: 0.1107 - accuracy: 0.9576 - val_loss: 0.1211 - val_accuracy: 0.9582
 - val_kappa:  0.9176

                Validation kappa did not improved from 0.9208342709556043
Epoch 14/30
389/389 [==============================] - 102s 261ms/step - loss: 0.1086 - accuracy: 0.9585 - val_loss: 0.1237 - val_accuracy: 0.9575
 - val_kappa:  0.9011

                Validation kappa did not improved from 0.9208342709556043
Epoch 15/30
389/389 [==============================] - 102s 262ms/step - loss: 0.1052 - accuracy: 0.9589 - val_loss: 0.1070 - val_accuracy: 0.9604
 - val_kappa:  0.9068

                Validation kappa did not improved from 0.9208342709556043
Epoch 16/30
389/389 [==============================] - 104s 268ms/step - loss: 0.1058 - accuracy: 0.9602 - val_loss: 0.1110 - val_accuracy: 0.9578
 - val_kappa:  0.9230

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/densenet.h5...
Epoch 17/30
389/389 [==============================] - 105s 271ms/step - loss: 0.1097 - accuracy: 0.9595 - val_loss: 0.1275 - val_accuracy: 0.9538
 - val_kappa:  0.8860

                Validation kappa did not improved from 0.9229616496760168
Epoch 18/30
389/389 [==============================] - 106s 273ms/step - loss: 0.1053 - accuracy: 0.9618 - val_loss: 0.1154 - val_accuracy: 0.9567
 - val_kappa:  0.8893

                Validation kappa did not improved from 0.9229616496760168
Epoch 19/30
389/389 [==============================] - 108s 277ms/step - loss: 0.1003 - accuracy: 0.9621 - val_loss: 0.1071 - val_accuracy: 0.9629
 - val_kappa:  0.9128

                Validation kappa did not improved from 0.9229616496760168
Epoch 20/30
389/389 [==============================] - 107s 275ms/step - loss: 0.0943 - accuracy: 0.9637 - val_loss: 0.1057 - val_accuracy: 0.9600
 - val_kappa:  0.9167

                Validation kappa did not improved from 0.9229616496760168
Epoch 21/30
389/389 [==============================] - 109s 279ms/step - loss: 0.1099 - accuracy: 0.9607 - val_loss: 0.1112 - val_accuracy: 0.9600
 - val_kappa:  0.9067

                Validation kappa did not improved from 0.9229616496760168
Epoch 22/30
389/389 [==============================] - 109s 280ms/step - loss: 0.0992 - accuracy: 0.9631 - val_loss: 0.1128 - val_accuracy: 0.9560
 - val_kappa:  0.8836

                Validation kappa did not improved from 0.9229616496760168
Epoch 23/30
389/389 [==============================] - 109s 279ms/step - loss: 0.0953 - accuracy: 0.9643 - val_loss: 0.1220 - val_accuracy: 0.9582
 - val_kappa:  0.9055

                Validation kappa did not improved from 0.9229616496760168
Epoch 24/30
389/389 [==============================] - 109s 280ms/step - loss: 0.0837 - accuracy: 0.9670 - val_loss: 0.1020 - val_accuracy: 0.9607
 - val_kappa:  0.9229

                Validation kappa did not improved from 0.9229616496760168
Epoch 25/30
389/389 [==============================] - 109s 279ms/step - loss: 0.0947 - accuracy: 0.9655 - val_loss: 0.1152 - val_accuracy: 0.9600
 - val_kappa:  0.9112

                Validation kappa did not improved from 0.9229616496760168
Epoch 26/30
389/389 [==============================] - 110s 282ms/step - loss: 0.0835 - accuracy: 0.9670 - val_loss: 0.1083 - val_accuracy: 0.9622
 - val_kappa:  0.9101

                Validation kappa did not improved from 0.9229616496760168
Epoch 27/30
389/389 [==============================] - 110s 284ms/step - loss: 0.0883 - accuracy: 0.9670 - val_loss: 0.1089 - val_accuracy: 0.9545
 - val_kappa:  0.9152

                Validation kappa did not improved from 0.9229616496760168
Epoch 28/30
389/389 [==============================] - 111s 285ms/step - loss: 0.0791 - accuracy: 0.9696 - val_loss: 0.1159 - val_accuracy: 0.9618
 - val_kappa:  0.9223

                Validation kappa did not improved from 0.9229616496760168
Epoch 29/30
389/389 [==============================] - 111s 284ms/step - loss: 0.0807 - accuracy: 0.9707 - val_loss: 0.1159 - val_accuracy: 0.9527
 - val_kappa:  0.9100

                Validation kappa did not improved from 0.9229616496760168
Epoch 30/30
389/389 [==============================] - 111s 285ms/step - loss: 0.0716 - accuracy: 0.9726 - val_loss: 0.1400 - val_accuracy: 0.9545
 - val_kappa:  0.8901
```

```
Validation kappa did not improved from 0.9229616496760168
```

In [ ]:
```
densenet = DenseNet()
densenet.load_weights("/content/drive/My Drive/models/densenet.h5")
result = densenet.evaluate(x_validation,labels_validation)
y_pred = densenet.predict(x_validation)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result[0],4),np.round(result[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

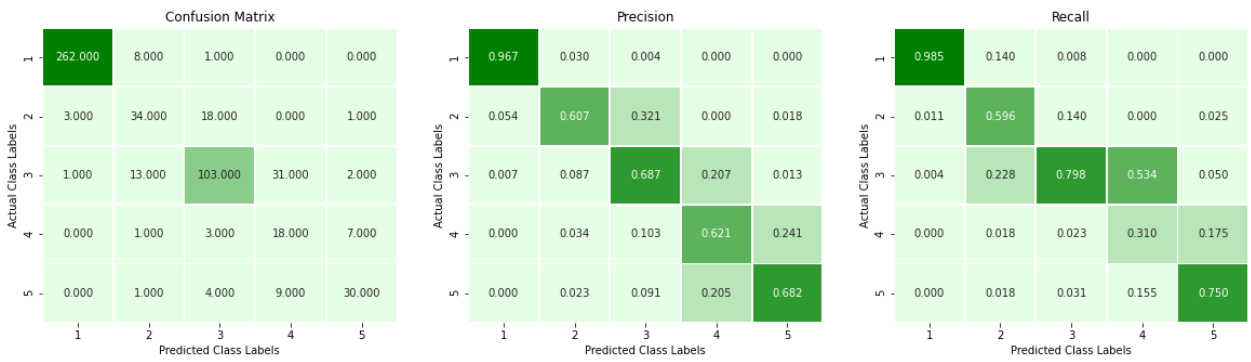
```
550/550 [==============================] - 10s 17ms/step
After running the model for 30 epochs we got loss = 0.111 Accuracy = 0.9578 kappa_score = 0.923 on validation data
```

In [ ]:
```
ytrain_densenet = densenet.predict(x_train)
ytrain_densenet = test_prediction(ytrain_densenet)
print("First five data points predictions in training:",ytrain_densenet[:5])
print("length of traindata prediction:",ytrain_densenet.shape,"\n")

yvalidation_densenet = densenet.predict(x_validation)
yvalidation_densenet = test_prediction(yvalidation_densenet)
print("First five data points predictions in validation:",yvalidation_densenet[:5])
print("length of validation data prediction:",yvalidation_densenet.shape,"\n")

ytest_densenet = densenet.predict(x_test)
ytest_densenet = test_prediction(ytest_densenet)
print("First five data points predictions in test:",ytest_densenet[:5])
print("length of test data prediction:",ytest_densenet.shape)
```

```
First five data points predictions in training: [0 1 0 2 3]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 1 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 3 3 2 2]
length of test data prediction: (1928,)
```

In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing instead.
  import pandas.util.testing as tm
```



In [ ]:
```
%load_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/densenet'
```

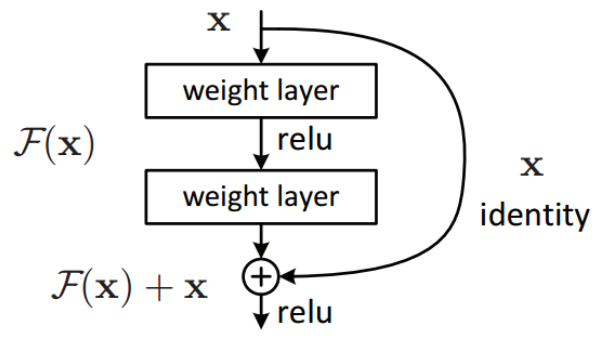
```
<IPython.core.display.Javascript object>
```

Tensorboard Visualization

Residual Network:

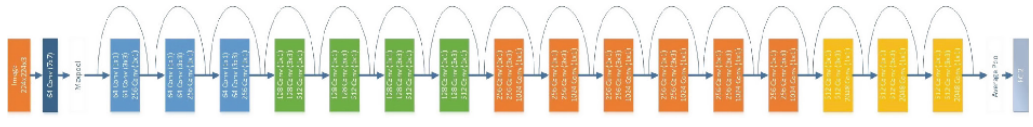
- Instead of directly stacking few layers to fit an underlying H(x) mapping.Explicitly let these layers to fit a residual mapping F(x).

Now,
F(x) = H(x) - x
H(x) = F(x) + x

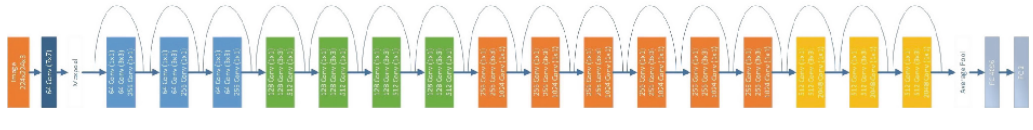


- Now this residual mapping(F(x)+x) are called skip connections which skips one or more layers.
- These connections perform identity mapping and there outputs are added to the outputs of the stacked layers.
- These connections add neighter extra parameters nor computational complexity.

Resnet-50 Architecture:



(a) Resnet50 Architecture



(b) Modified Resnet50 Architecture

In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def ResNet50_():
    '''This function is used for building a model architecture of pretrained Resnet50 on imagenet data set.'''
    resnet = ResNet50(weights='imagenet',include_top=False,layers=keras.layers,input_shape=(512,512,3))
    x = global_average_pooling_layer(resnet.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(resnet.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
resnet = ResNet50_()
resnet.summary()
```

```
Model: "Resnet-50"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
resnet50 (Functional)        (None, 16, 16, 2048)      23587712
_________________________________________________________________
global_average_pooling2d_21  (None, 2048)              0
_________________________________________________________________
dropout_25 (Dropout)         (None, 2048)              0
_________________________________________________________________
dense_35 (Dense)             (None, 5)                 10245
=================================================================
Total params: 23,597,957
Trainable params: 23,544,837
Non-trainable params: 53,120
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/resnet50')
kappa_metrics = Metrics('/content/drive/My Drive/models/resnet50.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
resnet = ResNet50_()
history = resnet.fit_generator(
                    data_generator.flow(x_train, labels_train, batch_size = 8),
                     steps_per_epoch=len(x_train) / 8,
                     epochs=30,
                     initial_epoch=0,
                     verbose=1,
                     validation_data=(x_validation, labels_validation),
                     validation_steps=len(x_validation) / 8,
                     callbacks=[kappa_metrics,tensorboard], class_weight = class_weights)
```

```
Downloading data from https://github.com/keras-team/keras-applications/releases/download/resnet/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5 (https://github.com/keras-team/keras-applications/releases/download/resnet/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5)
94773248/94765736 [==============================] - 4s 0us/step
Epoch 1/30
  3/389 [..............................] - ETA: 42:34 - loss: 0.8961 - accuracy: 0.4333

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (1.841238). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 86s 220ms/step - loss: 0.4265 - accuracy: 0.8000 - val_loss: 0.2152 - val_accuracy: 0.9200
 - val_kappa:  0.7633

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 2/30
389/389 [==============================] - 64s 165ms/step - loss: 0.2450 - accuracy: 0.9027 - val_loss: 0.1651 - val_accuracy: 0.9364
 - val_kappa:  0.8196

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 3/30
389/389 [==============================] - 64s 164ms/step - loss: 0.2063 - accuracy: 0.9179 - val_loss: 0.1475 - val_accuracy: 0.9422
 - val_kappa:  0.8489

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 4/30
389/389 [==============================] - 64s 164ms/step - loss: 0.1914 - accuracy: 0.9228 - val_loss: 0.1340 - val_accuracy: 0.9462
 - val_kappa:  0.8736

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 5/30
389/389 [==============================] - 63s 161ms/step - loss: 0.1791 - accuracy: 0.9266 - val_loss: 0.1336 - val_accuracy: 0.9455
 - val_kappa:  0.8720

               Validation kappa did not improved from 0.873614723362475
Epoch 6/30
389/389 [==============================] - 62s 159ms/step - loss: 0.1644 - accuracy: 0.9373 - val_loss: 0.1345 - val_accuracy: 0.9469
 - val_kappa:  0.8756

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 7/30
389/389 [==============================] - 62s 159ms/step - loss: 0.1613 - accuracy: 0.9371 - val_loss: 0.1384 - val_accuracy: 0.9465
 - val_kappa:  0.8831

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 8/30
389/389 [==============================] - 62s 159ms/step - loss: 0.1538 - accuracy: 0.9415 - val_loss: 0.1300 - val_accuracy: 0.9462
 - val_kappa:  0.8761

               Validation kappa did not improved from 0.8831338183271787
Epoch 9/30
389/389 [==============================] - 61s 156ms/step - loss: 0.1475 - accuracy: 0.9409 - val_loss: 0.1272 - val_accuracy: 0.9473
 - val_kappa:  0.8810

               Validation kappa did not improved from 0.8831338183271787
Epoch 10/30
389/389 [==============================] - 60s 155ms/step - loss: 0.1392 - accuracy: 0.9456 - val_loss: 0.1315 - val_accuracy: 0.9491
 - val_kappa:  0.8850

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 11/30
389/389 [==============================] - 60s 154ms/step - loss: 0.1376 - accuracy: 0.9462 - val_loss: 0.1323 - val_accuracy: 0.9462
 - val_kappa:  0.8859

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 12/30
389/389 [==============================] - 60s 154ms/step - loss: 0.1346 - accuracy: 0.9476 - val_loss: 0.1226 - val_accuracy: 0.9505
 - val_kappa:  0.8930

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 13/30
389/389 [==============================] - 60s 154ms/step - loss: 0.1288 - accuracy: 0.9489 - val_loss: 0.1204 - val_accuracy: 0.9538
 - val_kappa:  0.8926

               Validation kappa did not improved from 0.8930354851441351
Epoch 14/30
389/389 [==============================] - 59s 152ms/step - loss: 0.1259 - accuracy: 0.9510 - val_loss: 0.1141 - val_accuracy: 0.9571
 - val_kappa:  0.9049

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 15/30
389/389 [==============================] - 60s 154ms/step - loss: 0.1224 - accuracy: 0.9504 - val_loss: 0.1207 - val_accuracy: 0.9516
 - val_kappa:  0.8892

               Validation kappa did not improved from 0.9049075524397799
Epoch 16/30
389/389 [==============================] - 59s 152ms/step - loss: 0.1158 - accuracy: 0.9546 - val_loss: 0.1237 - val_accuracy: 0.9513
 - val_kappa:  0.8931

               Validation kappa did not improved from 0.9049075524397799
Epoch 17/30
389/389 [==============================] - 60s 155ms/step - loss: 0.1183 - accuracy: 0.9545 - val_loss: 0.1318 - val_accuracy: 0.9495
 - val_kappa:  0.9017

               Validation kappa did not improved from 0.9049075524397799
Epoch 18/30
389/389 [==============================] - 61s 157ms/step - loss: 0.1144 - accuracy: 0.9548 - val_loss: 0.1218 - val_accuracy: 0.9545
 - val_kappa:  0.9047

               Validation kappa did not improved from 0.9049075524397799
Epoch 19/30
389/389 [==============================] - 60s 154ms/step - loss: 0.1105 - accuracy: 0.9575 - val_loss: 0.1222 - val_accuracy: 0.9480
 - val_kappa:  0.8891

               Validation kappa did not improved from 0.9049075524397799
Epoch 20/30
389/389 [==============================] - 60s 154ms/step - loss: 0.1129 - accuracy: 0.9589 - val_loss: 0.1235 - val_accuracy: 0.9524
 - val_kappa:  0.8989

               Validation kappa did not improved from 0.9049075524397799
Epoch 21/30
389/389 [==============================] - 60s 153ms/step - loss: 0.1064 - accuracy: 0.9600 - val_loss: 0.1296 - val_accuracy: 0.9538
 - val_kappa:  0.9051

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 22/30
389/389 [==============================] - 60s 154ms/step - loss: 0.1029 - accuracy: 0.9602 - val_loss: 0.1246 - val_accuracy: 0.9535
 - val_kappa:  0.9005

               Validation kappa did not improved from 0.9050940579512938
Epoch 23/30
389/389 [==============================] - 60s 153ms/step - loss: 0.1005 - accuracy: 0.9601 - val_loss: 0.1183 - val_accuracy: 0.9556
 - val_kappa:  0.9064

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 24/30
389/389 [==============================] - 62s 159ms/step - loss: 0.1053 - accuracy: 0.9587 - val_loss: 0.1264 - val_accuracy: 0.9538
 - val_kappa:  0.8938

               Validation kappa did not improved from 0.9064485467006854
Epoch 25/30
389/389 [==============================] - 60s 154ms/step - loss: 0.0974 - accuracy: 0.9611 - val_loss: 0.1264 - val_accuracy: 0.9567
 - val_kappa:  0.9091

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet50.h5...
Epoch 26/30
389/389 [==============================] - 60s 155ms/step - loss: 0.0943 - accuracy: 0.9615 - val_loss: 0.1275 - val_accuracy: 0.9513
 - val_kappa:  0.8934

               Validation kappa did not improved from 0.9090632747079858
Epoch 27/30
389/389 [==============================] - 59s 153ms/step - loss: 0.0922 - accuracy: 0.9647 - val_loss: 0.1264 - val_accuracy: 0.9556
 - val_kappa:  0.8917

               Validation kappa did not improved from 0.9090632747079858
Epoch 28/30
389/389 [==============================] - 60s 153ms/step - loss: 0.0934 - accuracy: 0.9626 - val_loss: 0.1288 - val_accuracy: 0.9538
 - val_kappa:  0.9034

               Validation kappa did not improved from 0.9090632747079858
Epoch 29/30
389/389 [==============================] - 60s 153ms/step - loss: 0.0911 - accuracy: 0.9645 - val_loss: 0.1280 - val_accuracy: 0.9575
 - val_kappa:  0.9086

               Validation kappa did not improved from 0.9090632747079858
```

```
Epoch 30/30
389/389 [==============================] - 60s 153ms/step - loss: 0.0851 - accuracy: 0.9660 - val_loss: 0.1345 - val_accuracy: 0.9545
 - val_kappa:  0.9000

               Validation kappa did not improved from 0.9090632747079858
```

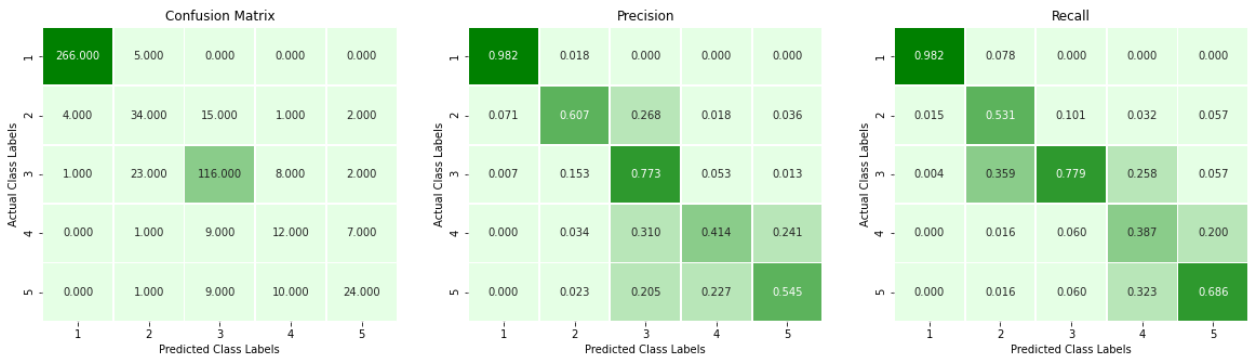
In [ ]:
```
resnet = ResNet50_()
resnet.load_weights("/content/drive/My Drive/models/resnet50.h5")
result = resnet.evaluate(x_validation,labels_validation)
y_pred = resnet.predict(x_validation)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result[0],4),np.round(result[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
550/550 [==============================] - 7s 13ms/step
After running the model for 30 epochs we got loss = 0.1264 Accuracy = 0.9567 kappa_score = 0.9091 on validation data
```

In [ ]:
```
ytrain_resnet = resnet.predict(x_train)
ytrain_resnet = test_prediction(ytrain_resnet)
print("First five data points predictions in training:",ytrain_resnet[:5])
print("length of traindata prediction:",ytrain_resnet.shape,"\n")

yvalidation_resnet = resnet.predict(x_validation)
yvalidation_resnet = test_prediction(yvalidation_resnet)
print("First five data points predictions in validation:",yvalidation_resnet[:5])
print("length of validation data prediction:",yvalidation_resnet.shape,"\n")

ytest_resnet = resnet.predict(x_test)
ytest_resnet = test_prediction(ytest_resnet)
print("First five data points predictions in test:",ytest_resnet[:5])
print("length of test data prediction:",ytest_resnet.shape)
```

```
First five data points predictions in training: [0 2 0 2 1]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 0 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 2 3 2 3]
length of test data prediction: (1928,)
```

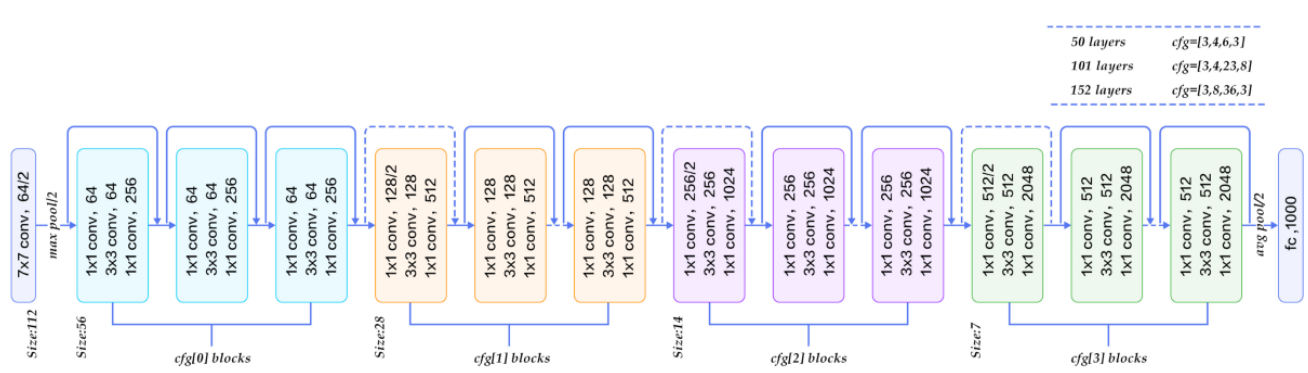
In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/resnet50'
```

Tensorboard visualization:

Resnet-152 Architecture:



In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def ResNet152_():
    '''This function is used for building a model architecture of pretrained Resnet152 on imagenet data set.'''
    resnet = ResNet152(weights='imagenet',include_top=False,layers=keras.layers,input_shape=(512,512,3))
    x = global_average_pooling_layer(resnet.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(resnet.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
resnet = ResNet152_()
resnet.summary()
```

```
Model: "sequential_7"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
resnet152 (Functional)       (None, 16, 16, 2048)      58370944
_________________________________________________________________
global_average_pooling2d_9 ( (None, 2048)              0
_________________________________________________________________
dropout_13 (Dropout)         (None, 2048)              0
_________________________________________________________________
dense_23 (Dense)             (None, 5)                 10245
=================================================================
Total params: 58,381,189
Trainable params: 58,229,765
Non-trainable params: 151,424
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/resnet152')
kappa_metrics = Metrics('/content/drive/My Drive/models/resnet152.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
resnet = ResNet152_()
history = resnet.fit_generator(
                    data_generator.flow(x_train, labels_train, batch_size = 8),
                     steps_per_epoch=len(x_train) / 8,
                     epochs=30,
                     initial_epoch=0,
                     verbose=1,
                     validation_data=(x_validation, labels_validation),
                     validation_steps=len(x_validation) / 8,
                     callbacks=[kappa_metrics,tensorboard], class_weight = class_weights)
```

```
Epoch 1/30
  2/389 [..............................] - ETA: 3:43:34 - loss: 0.9010 - accuracy: 0.4250

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (0.943490). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 219s 564ms/step - loss: 0.3880 - accuracy: 0.8213 - val_loss: 0.1875 - val_accuracy: 0.9258
 - val_kappa:  0.7848

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 2/30
389/389 [==============================] - 144s 371ms/step - loss: 0.2126 - accuracy: 0.9134 - val_loss: 0.1499 - val_accuracy: 0.9385
 - val_kappa:  0.8364

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 3/30
389/389 [==============================] - 145s 373ms/step - loss: 0.1854 - accuracy: 0.9248 - val_loss: 0.1398 - val_accuracy: 0.9440
 - val_kappa:  0.8654

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 4/30
389/389 [==============================] - 146s 374ms/step - loss: 0.1720 - accuracy: 0.9308 - val_loss: 0.1308 - val_accuracy: 0.9480
 - val_kappa:  0.8818

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 5/30
389/389 [==============================] - 145s 374ms/step - loss: 0.1580 - accuracy: 0.9368 - val_loss: 0.1250 - val_accuracy: 0.9516
 - val_kappa:  0.8890

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 6/30
389/389 [==============================] - 146s 375ms/step - loss: 0.1574 - accuracy: 0.9406 - val_loss: 0.1350 - val_accuracy: 0.9487
 - val_kappa:  0.8825

                Validation kappa did not improved from 0.889012610651064
Epoch 7/30
389/389 [==============================] - 144s 370ms/step - loss: 0.1486 - accuracy: 0.9389 - val_loss: 0.1232 - val_accuracy: 0.9527
 - val_kappa:  0.8817

                Validation kappa did not improved from 0.889012610651064
Epoch 8/30
389/389 [==============================] - 144s 370ms/step - loss: 0.1355 - accuracy: 0.9492 - val_loss: 0.1179 - val_accuracy: 0.9585
 - val_kappa:  0.8931

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 9/30
389/389 [==============================] - 144s 370ms/step - loss: 0.1276 - accuracy: 0.9502 - val_loss: 0.1235 - val_accuracy: 0.9516
 - val_kappa:  0.9003

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 10/30
389/389 [==============================] - 145s 372ms/step - loss: 0.1303 - accuracy: 0.9488 - val_loss: 0.1162 - val_accuracy: 0.9516
 - val_kappa:  0.8966

                Validation kappa did not improved from 0.9003002509769674
Epoch 11/30
389/389 [==============================] - 142s 365ms/step - loss: 0.1214 - accuracy: 0.9538 - val_loss: 0.1210 - val_accuracy: 0.9593
 - val_kappa:  0.8990

                Validation kappa did not improved from 0.9003002509769674
Epoch 12/30
389/389 [==============================] - 144s 370ms/step - loss: 0.1196 - accuracy: 0.9539 - val_loss: 0.1076 - val_accuracy: 0.9593
 - val_kappa:  0.9024

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 13/30
389/389 [==============================] - 144s 370ms/step - loss: 0.1128 - accuracy: 0.9548 - val_loss: 0.1169 - val_accuracy: 0.9567
 - val_kappa:  0.9008

                Validation kappa did not improved from 0.9023660982584165
Epoch 14/30
389/389 [==============================] - 143s 368ms/step - loss: 0.1058 - accuracy: 0.9585 - val_loss: 0.1264 - val_accuracy: 0.9549
 - val_kappa:  0.8986

                Validation kappa did not improved from 0.9023660982584165
Epoch 15/30
389/389 [==============================] - 142s 365ms/step - loss: 0.1088 - accuracy: 0.9584 - val_loss: 0.1194 - val_accuracy: 0.9549
 - val_kappa:  0.9013

                Validation kappa did not improved from 0.9023660982584165
Epoch 16/30
389/389 [==============================] - 144s 369ms/step - loss: 0.1094 - accuracy: 0.9571 - val_loss: 0.1246 - val_accuracy: 0.9538
 - val_kappa:  0.9084

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 17/30
389/389 [==============================] - 145s 372ms/step - loss: 0.1025 - accuracy: 0.9600 - val_loss: 0.1089 - val_accuracy: 0.9607
 - val_kappa:  0.9094

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 18/30
389/389 [==============================] - 145s 372ms/step - loss: 0.0979 - accuracy: 0.9614 - val_loss: 0.1141 - val_accuracy: 0.9560
 - val_kappa:  0.9076

                Validation kappa did not improved from 0.9094287292937391
Epoch 19/30
389/389 [==============================] - 143s 368ms/step - loss: 0.0938 - accuracy: 0.9645 - val_loss: 0.1118 - val_accuracy: 0.9622
 - val_kappa:  0.9109

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 20/30
389/389 [==============================] - 145s 372ms/step - loss: 0.0920 - accuracy: 0.9634 - val_loss: 0.1190 - val_accuracy: 0.9615
 - val_kappa:  0.9089

                Validation kappa did not improved from 0.9108914432640454
Epoch 21/30
389/389 [==============================] - 143s 367ms/step - loss: 0.0856 - accuracy: 0.9654 - val_loss: 0.1222 - val_accuracy: 0.9560
 - val_kappa:  0.8993

                Validation kappa did not improved from 0.9108914432640454
Epoch 22/30
389/389 [==============================] - 144s 371ms/step - loss: 0.0845 - accuracy: 0.9659 - val_loss: 0.1175 - val_accuracy: 0.9585
 - val_kappa:  0.9077

                Validation kappa did not improved from 0.9108914432640454
Epoch 23/30
389/389 [==============================] - 144s 369ms/step - loss: 0.0819 - accuracy: 0.9683 - val_loss: 0.1297 - val_accuracy: 0.9567
 - val_kappa:  0.9054

                Validation kappa did not improved from 0.9108914432640454
Epoch 24/30
389/389 [==============================] - 144s 371ms/step - loss: 0.0830 - accuracy: 0.9684 - val_loss: 0.1194 - val_accuracy: 0.9607
 - val_kappa:  0.9139

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 25/30
389/389 [==============================] - 145s 372ms/step - loss: 0.0817 - accuracy: 0.9690 - val_loss: 0.1263 - val_accuracy: 0.9567
 - val_kappa:  0.8993

                Validation kappa did not improved from 0.9138537748753751
Epoch 26/30
389/389 [==============================] - 143s 367ms/step - loss: 0.0726 - accuracy: 0.9720 - val_loss: 0.1188 - val_accuracy: 0.9567
 - val_kappa:  0.9070

                Validation kappa did not improved from 0.9138537748753751
Epoch 27/30
389/389 [==============================] - 143s 367ms/step - loss: 0.0727 - accuracy: 0.9710 - val_loss: 0.1164 - val_accuracy: 0.9585
 - val_kappa:  0.9146

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/resnet152.h5...
Epoch 28/30
389/389 [==============================] - 145s 373ms/step - loss: 0.0684 - accuracy: 0.9724 - val_loss: 0.1382 - val_accuracy: 0.9516
 - val_kappa:  0.8975

                Validation kappa did not improved from 0.9145890230847753
Epoch 29/30
389/389 [==============================] - 144s 370ms/step - loss: 0.0718 - accuracy: 0.9729 - val_loss: 0.1356 - val_accuracy: 0.9538
 - val_kappa:  0.9004

                Validation kappa did not improved from 0.9145890230847753
Epoch 30/30
389/389 [==============================] - 144s 369ms/step - loss: 0.0703 - accuracy: 0.9723 - val_loss: 0.1292 - val_accuracy: 0.9571
 - val_kappa:  0.9066
```

```
Validation kappa did not improved from 0.9145890230847753
```

In [ ]:
```
resnet = ResNet152_()
resnet.load_weights("/content/drive/My Drive/models/resnet152.h5")
result = resnet.evaluate(x_validation,labels_validation)
y_pred = resnet.predict(x_validation)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result[0],4),np.round(result[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

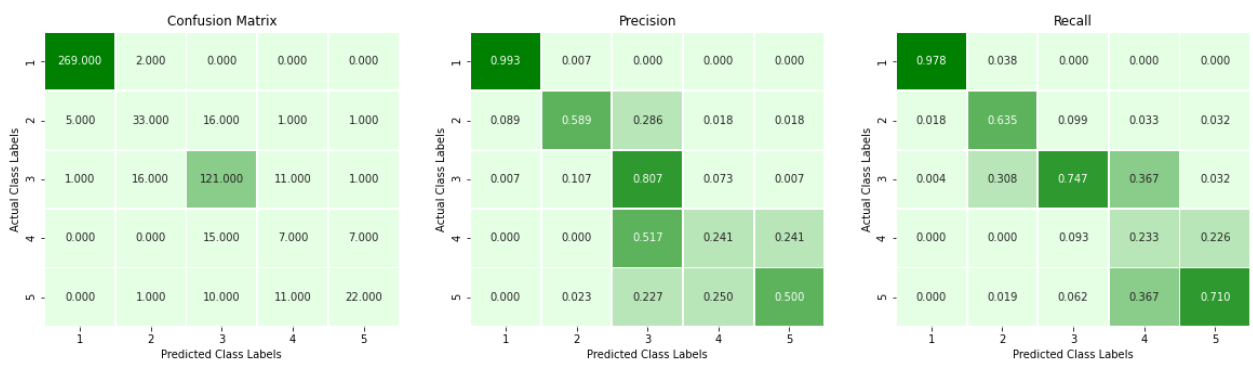
```
550/550 [==============================] - 19s 34ms/step
After running the model for 30 epochs we got loss = 0.1164 Accuracy = 0.9585 kappa_score = 0.9146 on validation data
```

In [ ]:
```
ytrain_resnet1 = resnet.predict(x_train)
ytrain_resnet1 = test_prediction(ytrain_resnet1)
print("First five data points predictions in training:",ytrain_resnet1[:5])
print("length of traindata prediction:",ytrain_resnet1.shape,"\n")

yvalidation_resnet1 = resnet.predict(x_validation)
yvalidation_resnet1 = test_prediction(yvalidation_resnet1)
print("First five data points predictions in validation:",yvalidation_resnet1[:5])
print("length of validation data prediction:",yvalidation_resnet1.shape,"\n")

ytest_resnet1 = resnet.predict(x_test)
ytest_resnet1 = test_prediction(ytest_resnet1)
print("First five data points predictions in test:",ytest_resnet1[:5])
print("length of test data prediction:",ytest_resnet1.shape)
```

```
First five data points predictions in training: [0 1 0 2 2]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 0 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 2 2 2 2]
length of test data prediction: (1928,)
```

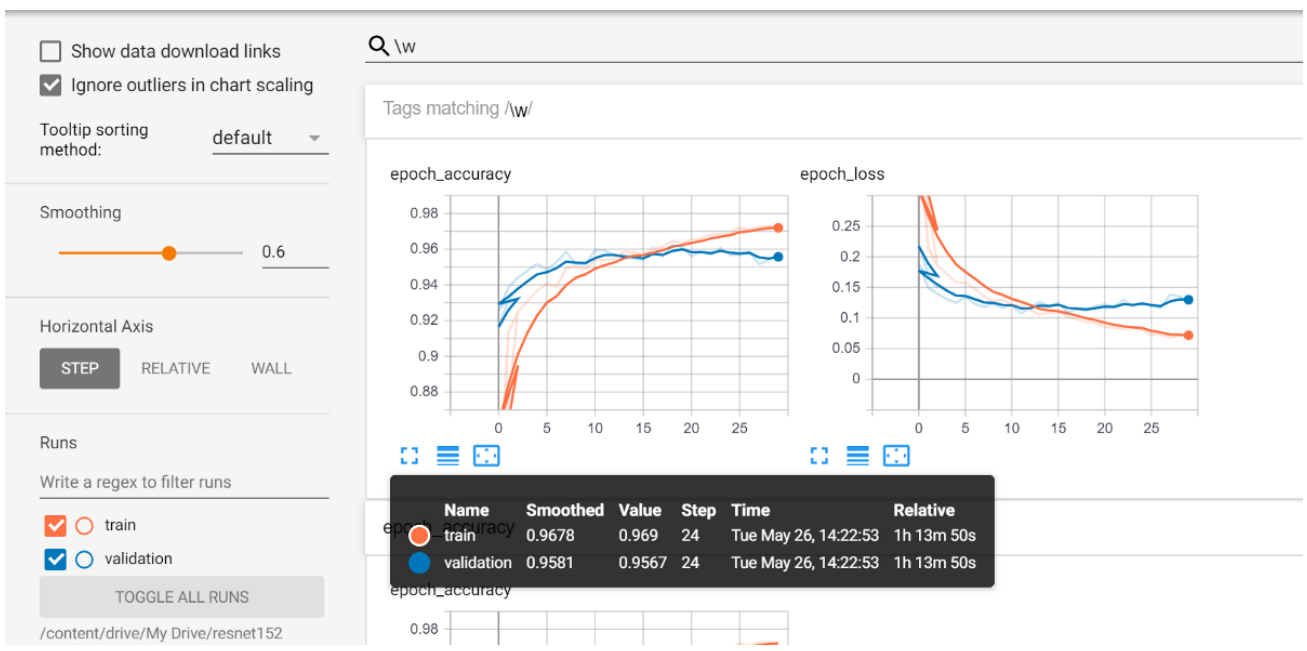
In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing instead.
  import pandas.util.testing as tm
```



In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/resnet152'
```

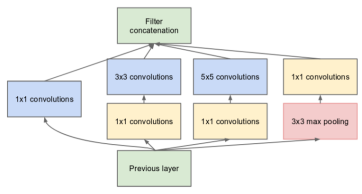
Tensorboard Visualization:



Inception Networks:

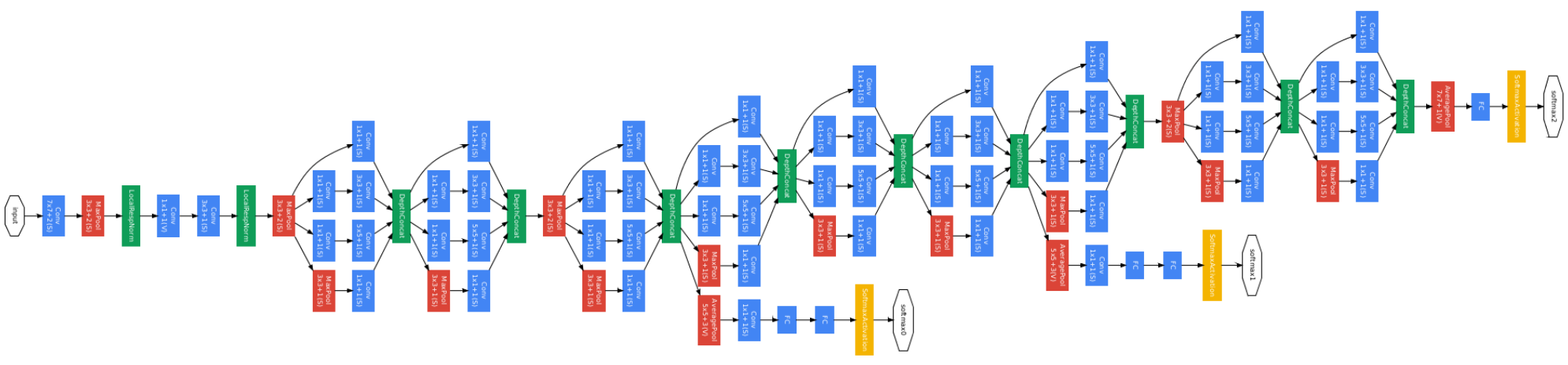
- Inception model allows us to apply multiple filters(1x1, 3x3, 5x5) and even max pooling layer on the same level and concatenate the results. Thus the network essentially would get a bit 'wider' rather than 'deeper'.

Inception module:



(b) Inception module with dimension reductions

Inception v3 Architecture:



In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def inceptionv3_():
    '''This function is used for building a model architecture of pretrained InceptionV3 on imagenet data set.'''
    inceptionv3 = InceptionV3(weights = 'imagenet', include_top = False, input_shape = (512,512,3))
    x = global_average_pooling_layer(inceptionv3.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(inceptionv3.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
inceptionv3 = inceptionv3_()
inceptionv3.summary()
```

```
Model: "inception-v3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
inception_v3 (Functional)    (None, 14, 14, 2048)      21802784
_________________________________________________________________
global_average_pooling2d_11  (None, 2048)              0
_________________________________________________________________
dropout_15 (Dropout)         (None, 2048)              0
_________________________________________________________________
dense_25 (Dense)             (None, 5)                 10245
=================================================================
Total params: 21,813,029
Trainable params: 21,778,597
Non-trainable params: 34,432
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/inceptionv3')
kappa_metrics = Metrics('/content/drive/My Drive/models/inceptionv3.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
inceptionv3 = inceptionv3_()
history = inceptionv3.fit_generator(
                    data_generator.flow(x_train, labels_train, batch_size = 8),
                     steps_per_epoch=len(x_train) / 8,
                     epochs=30,
                     initial_epoch=0,
                     verbose=1,
                     validation_data=(x_validation, labels_validation),
                     validation_steps=len(x_validation) / 8,
                     callbacks=[kappa_metrics,tensorboard],class_weight = class_weights)
```

```
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.5/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5 (https://github.com/fchollet/deep-learning-models/releases/download/v0.5/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5)
87916544/87910968 [==============================] - 3s 0us/step
Epoch 1/30
  3/389 [..............................] - ETA: 58:11 - loss: 0.8915 - accuracy: 0.3583

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (3.077288). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 101s 258ms/step - loss: 0.5048 - accuracy: 0.7531 - val_loss: 0.2687 - val_accuracy: 0.9033
 - val_kappa:  0.6694

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 2/30
389/389 [==============================] - 71s 183ms/step - loss: 0.2789 - accuracy: 0.8956 - val_loss: 0.2201 - val_accuracy: 0.9116
 - val_kappa:  0.6890

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 3/30
389/389 [==============================] - 71s 183ms/step - loss: 0.2328 - accuracy: 0.9093 - val_loss: 0.2010 - val_accuracy: 0.9175
 - val_kappa:  0.7267

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 4/30
389/389 [==============================] - 72s 185ms/step - loss: 0.2178 - accuracy: 0.9135 - val_loss: 0.1808 - val_accuracy: 0.9244
 - val_kappa:  0.7676

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 5/30
389/389 [==============================] - 72s 186ms/step - loss: 0.2094 - accuracy: 0.9168 - val_loss: 0.1733 - val_accuracy: 0.9265
 - val_kappa:  0.7663

                Validation kappa did not improved from 0.7675974979177811
Epoch 6/30
389/389 [==============================] - 71s 182ms/step - loss: 0.1955 - accuracy: 0.9236 - val_loss: 0.1620 - val_accuracy: 0.9320
 - val_kappa:  0.8013

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 7/30
389/389 [==============================] - 71s 182ms/step - loss: 0.1882 - accuracy: 0.9256 - val_loss: 0.1651 - val_accuracy: 0.9269
 - val_kappa:  0.7757

                Validation kappa did not improved from 0.8012508139593127
Epoch 8/30
389/389 [==============================] - 70s 181ms/step - loss: 0.1826 - accuracy: 0.9280 - val_loss: 0.1515 - val_accuracy: 0.9400
 - val_kappa:  0.8386

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 9/30
389/389 [==============================] - 70s 180ms/step - loss: 0.1799 - accuracy: 0.9314 - val_loss: 0.1529 - val_accuracy: 0.9349
 - val_kappa:  0.8237

                Validation kappa did not improved from 0.8386087092609257
Epoch 10/30
389/389 [==============================] - 70s 180ms/step - loss: 0.1655 - accuracy: 0.9344 - val_loss: 0.1451 - val_accuracy: 0.9371
 - val_kappa:  0.8318

                Validation kappa did not improved from 0.8386087092609257
Epoch 11/30
389/389 [==============================] - 71s 182ms/step - loss: 0.1652 - accuracy: 0.9348 - val_loss: 0.1360 - val_accuracy: 0.9473
 - val_kappa:  0.8739

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 12/30
389/389 [==============================] - 72s 185ms/step - loss: 0.1602 - accuracy: 0.9375 - val_loss: 0.1344 - val_accuracy: 0.9473
 - val_kappa:  0.8735

                Validation kappa did not improved from 0.8739159816518147
Epoch 13/30
389/389 [==============================] - 71s 184ms/step - loss: 0.1582 - accuracy: 0.9378 - val_loss: 0.1331 - val_accuracy: 0.9484
 - val_kappa:  0.8793

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 14/30
389/389 [==============================] - 73s 188ms/step - loss: 0.1516 - accuracy: 0.9415 - val_loss: 0.1265 - val_accuracy: 0.9516
 - val_kappa:  0.8947

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 15/30
389/389 [==============================] - 74s 189ms/step - loss: 0.1457 - accuracy: 0.9441 - val_loss: 0.1249 - val_accuracy: 0.9495
 - val_kappa:  0.8896

                Validation kappa did not improved from 0.8947042716433171
Epoch 16/30
389/389 [==============================] - 72s 186ms/step - loss: 0.1393 - accuracy: 0.9459 - val_loss: 0.1296 - val_accuracy: 0.9509
 - val_kappa:  0.8998

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 17/30
389/389 [==============================] - 72s 186ms/step - loss: 0.1392 - accuracy: 0.9443 - val_loss: 0.1211 - val_accuracy: 0.9538
 - val_kappa:  0.8995

                Validation kappa did not improved from 0.8998080913368338
Epoch 18/30
389/389 [==============================] - 71s 183ms/step - loss: 0.1414 - accuracy: 0.9455 - val_loss: 0.1262 - val_accuracy: 0.9509
 - val_kappa:  0.8904

                Validation kappa did not improved from 0.8998080913368338
Epoch 19/30
389/389 [==============================] - 71s 181ms/step - loss: 0.1274 - accuracy: 0.9492 - val_loss: 0.1227 - val_accuracy: 0.9538
 - val_kappa:  0.8914

                Validation kappa did not improved from 0.8998080913368338
Epoch 20/30
389/389 [==============================] - 70s 179ms/step - loss: 0.1282 - accuracy: 0.9499 - val_loss: 0.1214 - val_accuracy: 0.9524
 - val_kappa:  0.8912

                Validation kappa did not improved from 0.8998080913368338
Epoch 21/30
389/389 [==============================] - 69s 178ms/step - loss: 0.1286 - accuracy: 0.9500 - val_loss: 0.1204 - val_accuracy: 0.9545
 - val_kappa:  0.9014

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/inceptionv3.h5...
Epoch 22/30
389/389 [==============================] - 71s 182ms/step - loss: 0.1269 - accuracy: 0.9501 - val_loss: 0.1350 - val_accuracy: 0.9487
 - val_kappa:  0.8687

                Validation kappa did not improved from 0.9014283625994477
Epoch 23/30
389/389 [==============================] - 70s 180ms/step - loss: 0.1270 - accuracy: 0.9508 - val_loss: 0.1177 - val_accuracy: 0.9545
 - val_kappa:  0.8963

                Validation kappa did not improved from 0.9014283625994477
Epoch 24/30
389/389 [==============================] - 70s 179ms/step - loss: 0.1213 - accuracy: 0.9517 - val_loss: 0.1180 - val_accuracy: 0.9545
 - val_kappa:  0.8936

                Validation kappa did not improved from 0.9014283625994477
Epoch 25/30
389/389 [==============================] - 70s 180ms/step - loss: 0.1216 - accuracy: 0.9541 - val_loss: 0.1166 - val_accuracy: 0.9545
 - val_kappa:  0.8938

                Validation kappa did not improved from 0.9014283625994477
Epoch 26/30
389/389 [==============================] - 70s 180ms/step - loss: 0.1246 - accuracy: 0.9496 - val_loss: 0.1194 - val_accuracy: 0.9535
 - val_kappa:  0.8975

                Validation kappa did not improved from 0.9014283625994477
Epoch 27/30
389/389 [==============================] - 71s 182ms/step - loss: 0.1191 - accuracy: 0.9533 - val_loss: 0.1189 - val_accuracy: 0.9524
 - val_kappa:  0.8826

                Validation kappa did not improved from 0.9014283625994477
Epoch 28/30
389/389 [==============================] - 71s 181ms/step - loss: 0.1117 - accuracy: 0.9555 - val_loss: 0.1187 - val_accuracy: 0.9545
 - val_kappa:  0.8935

                Validation kappa did not improved from 0.9014283625994477
Epoch 29/30
389/389 [==============================] - 71s 182ms/step - loss: 0.1138 - accuracy: 0.9551 - val_loss: 0.1167 - val_accuracy: 0.9549
 - val_kappa:  0.8990

                Validation kappa did not improved from 0.9014283625994477
```

```
Epoch 30/30
389/389 [==============================] - 70s 181ms/step - loss: 0.1165 - accuracy: 0.9553 - val_loss: 0.1207 - val_accuracy: 0.9527
 - val_kappa:  0.8945

                Validation kappa did not improved from 0.9014283625994477
```

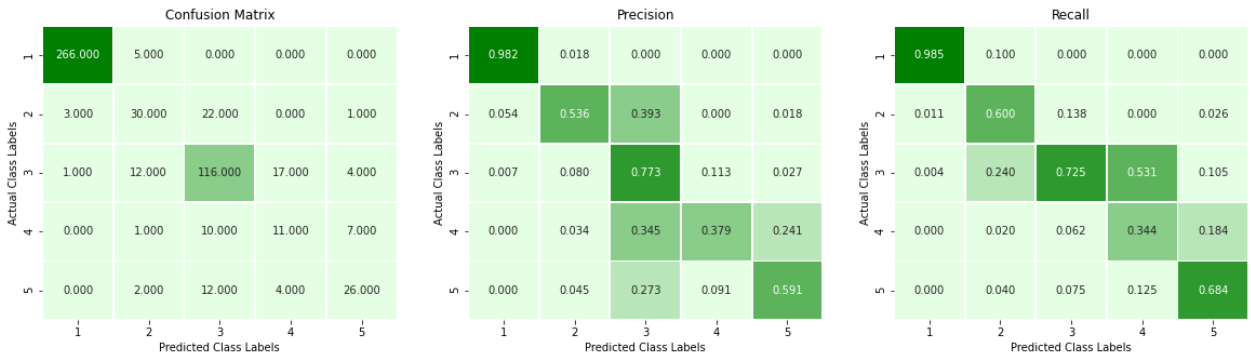
In [ ]:
```
inceptionv3 = inceptionv3_()
inceptionv3.load_weights("/content/drive/My Drive/models/inceptionv3.h5")
result1 = inceptionv3.evaluate(x_validation,labels_validation, verbose = 2)
y_pred = inceptionv3.predict(x_validation, batch_size = 8)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result1[0],4),np.round(result1[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
After running the model for 30 epochs we got loss = 0.1204 Accuracy = 0.9545 kappa_score = 0.9014 on validation data
```

In [ ]:
```
ytrain_inception = inceptionv3.predict(x_train)
ytrain_inception = test_prediction(ytrain_inception)
print("First five data points predictions in training:",ytrain_inception[:5])
print("length of traindata prediction:",ytrain_inception.shape,"\n")

yvalidation_inception = inceptionv3.predict(x_validation)
yvalidation_inception = test_prediction(yvalidation_inception)
print("First five data points predictions in validation:",yvalidation_inception[:5])
print("length of validation data prediction:",yvalidation_inception.shape,"\n")

ytest_inception = inceptionv3.predict(x_test)
ytest_inception = test_prediction(ytest_inception)
print("First five data points predictions in test:",ytest_inception[:5])
print("length of test data prediction:",ytest_inception.shape)
```

```
First five data points predictions in training: [0 2 0 2 2]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 1 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 2 2 1 3]
length of test data prediction: (1928,)
```

In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



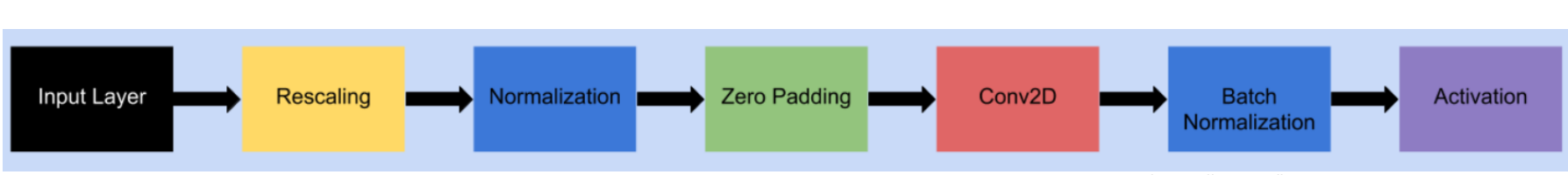
In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/inceptionv3'
```

Tensorboard Visualization:

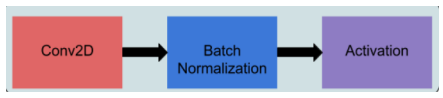
In [ ]:
```
pip install efficientnet
```

EfficientNet Networks:

- It starts with stem.

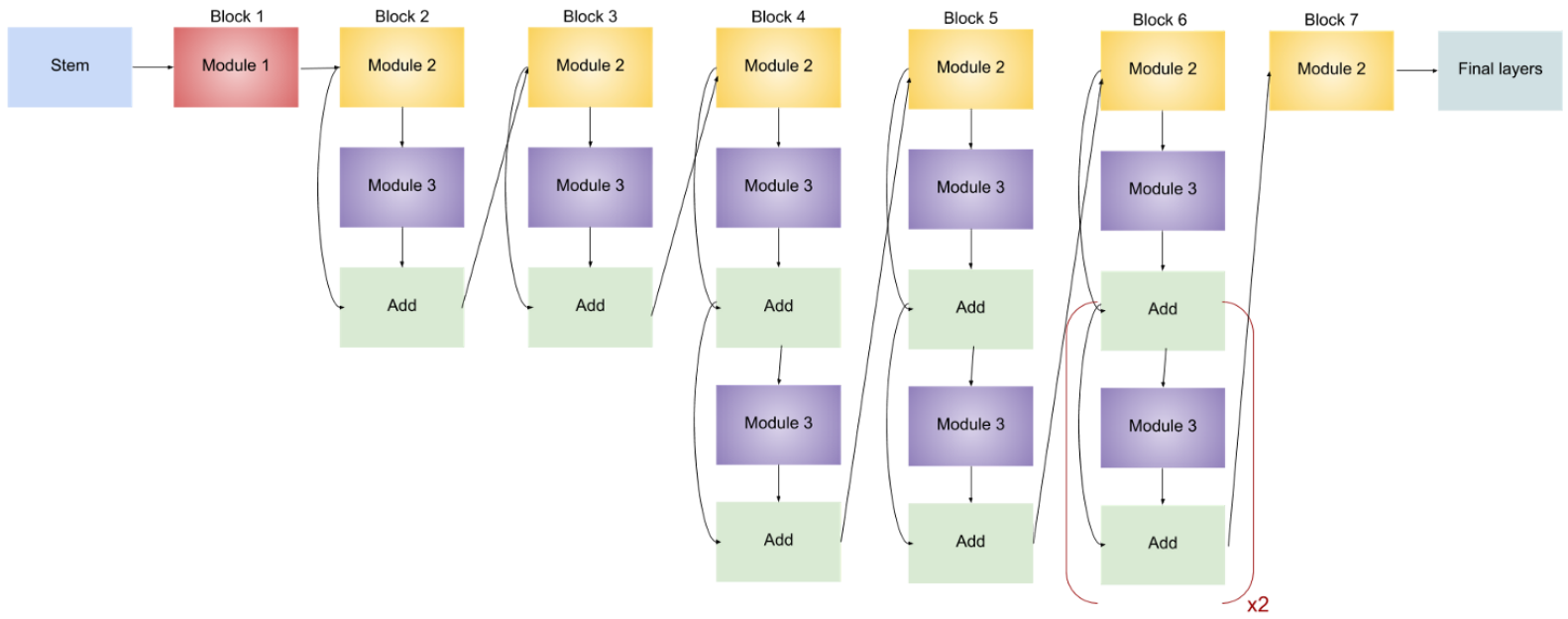


- Generally, the models are made too wide, deep, or with a very high resolution. Increasing these characteristics helps the model initially but it quickly saturates and the model made just has more parameters and is therefore not efficient. In EfficientNet they are scaled in a more principled way i.e. gradually everything is increased.
- The architecture consists of 7 blocks. These blocks further have a varying number of sub-blocks whose number is increased.
- Finally ends with Final layers.



Read more about efficientnet here (https://towardsdatascience.com/complete-architectural-details-of-all-efficientnet-models-5fd5b736142).

EfficientNet-B0 Architecture:



In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def efficientnet_b0():
    '''This function is used for building a model architecture of pretrained EfficientB0 on imagenet data set.'''
    efficientnet_ = EfficientNetB0(weights = 'imagenet',include_top = False, input_shape = (512,512,3))
    x = global_average_pooling_layer(efficientnet_.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(efficientnet_.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
efficientnet_ = efficientnet_b0()
efficientnet_.summary()
```

```
Model: "Efficientnet-B0"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnet-b0 (Functional) (None, 16, 16, 1280)      4049564
_________________________________________________________________
global_average_pooling2d_13  (None, 1280)              0
_________________________________________________________________
dropout_17 (Dropout)         (None, 1280)              0
_________________________________________________________________
dense_27 (Dense)             (None, 5)                 6405
=================================================================
Total params: 4,055,969
Trainable params: 4,013,953
Non-trainable params: 42,016
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/efficientnet_b0')
kappa_metrics = Metrics('/content/drive/My Drive/models/efficientnet_b0.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
efficientnet_ = efficientnet_b0()
history = efficientnet_.fit_generator(
                   data_generator.flow(x_train, labels_train, batch_size = 8),
                    steps_per_epoch=len(x_train) / 8,
                    epochs=30,
                    initial_epoch=0,
                    verbose=1,
                    validation_data=(x_validation, labels_validation),
                    validation_steps=len(x_validation) / 8,
                    callbacks=[kappa_metrics,tensorboard], class_weight = class_weights)
```

```
Epoch 1/30
  3/389 [..............................] - ETA: 43:21 - loss: 0.6731 - accuracy: 0.5833

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (1.638426). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 89s 229ms/step - loss: 0.3607 - accuracy: 0.8567 - val_loss: 0.2652 - val_accuracy: 0.8876
 - val_kappa:  0.5778

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 2/30
389/389 [==============================] - 67s 173ms/step - loss: 0.2073 - accuracy: 0.9193 - val_loss: 0.2083 - val_accuracy: 0.9102
 - val_kappa:  0.6983

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 3/30
389/389 [==============================] - 67s 171ms/step - loss: 0.1832 - accuracy: 0.9271 - val_loss: 0.1813 - val_accuracy: 0.9251
 - val_kappa:  0.7688

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 4/30
389/389 [==============================] - 67s 171ms/step - loss: 0.1697 - accuracy: 0.9328 - val_loss: 0.1733 - val_accuracy: 0.9342
 - val_kappa:  0.8310

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 5/30
389/389 [==============================] - 67s 171ms/step - loss: 0.1591 - accuracy: 0.9378 - val_loss: 0.1688 - val_accuracy: 0.9356
 - val_kappa:  0.8439

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 6/30
389/389 [==============================] - 67s 172ms/step - loss: 0.1451 - accuracy: 0.9407 - val_loss: 0.1521 - val_accuracy: 0.9371
 - val_kappa:  0.8675

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 7/30
389/389 [==============================] - 66s 171ms/step - loss: 0.1405 - accuracy: 0.9451 - val_loss: 0.1368 - val_accuracy: 0.9458
 - val_kappa:  0.8799

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 8/30
389/389 [==============================] - 66s 171ms/step - loss: 0.1345 - accuracy: 0.9458 - val_loss: 0.1450 - val_accuracy: 0.9444
 - val_kappa:  0.8677

                Validation kappa did not improved from 0.8799119767661377
Epoch 9/30
389/389 [==============================] - 67s 172ms/step - loss: 0.1265 - accuracy: 0.9511 - val_loss: 0.1188 - val_accuracy: 0.9560
 - val_kappa:  0.8949

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 10/30
389/389 [==============================] - 66s 171ms/step - loss: 0.1272 - accuracy: 0.9507 - val_loss: 0.1204 - val_accuracy: 0.9549
 - val_kappa:  0.8914

                Validation kappa did not improved from 0.8948844090076056
Epoch 11/30
389/389 [==============================] - 66s 168ms/step - loss: 0.1268 - accuracy: 0.9521 - val_loss: 0.1192 - val_accuracy: 0.9564
 - val_kappa:  0.8942

                Validation kappa did not improved from 0.8948844090076056
Epoch 12/30
389/389 [==============================] - 65s 168ms/step - loss: 0.1254 - accuracy: 0.9508 - val_loss: 0.1326 - val_accuracy: 0.9527
 - val_kappa:  0.8904

                Validation kappa did not improved from 0.8948844090076056
Epoch 13/30
389/389 [==============================] - 66s 169ms/step - loss: 0.1226 - accuracy: 0.9534 - val_loss: 0.1207 - val_accuracy: 0.9545
 - val_kappa:  0.8998

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 14/30
389/389 [==============================] - 66s 171ms/step - loss: 0.1122 - accuracy: 0.9560 - val_loss: 0.1295 - val_accuracy: 0.9553
 - val_kappa:  0.8970

                Validation kappa did not improved from 0.8997783473434541
Epoch 15/30
389/389 [==============================] - 65s 168ms/step - loss: 0.1160 - accuracy: 0.9554 - val_loss: 0.1196 - val_accuracy: 0.9589
 - val_kappa:  0.9015

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 16/30
389/389 [==============================] - 65s 167ms/step - loss: 0.1089 - accuracy: 0.9583 - val_loss: 0.1284 - val_accuracy: 0.9553
 - val_kappa:  0.8978

                Validation kappa did not improved from 0.9015136906035902
Epoch 17/30
389/389 [==============================] - 66s 169ms/step - loss: 0.1043 - accuracy: 0.9576 - val_loss: 0.1293 - val_accuracy: 0.9538
 - val_kappa:  0.9001

                Validation kappa did not improved from 0.9015136906035902
Epoch 18/30
389/389 [==============================] - 66s 170ms/step - loss: 0.0978 - accuracy: 0.9603 - val_loss: 0.1260 - val_accuracy: 0.9564
 - val_kappa:  0.9143

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 19/30
389/389 [==============================] - 65s 168ms/step - loss: 0.0992 - accuracy: 0.9596 - val_loss: 0.1264 - val_accuracy: 0.9513
 - val_kappa:  0.8974

                Validation kappa did not improved from 0.9142642803558032
Epoch 20/30
389/389 [==============================] - 65s 168ms/step - loss: 0.0929 - accuracy: 0.9641 - val_loss: 0.1140 - val_accuracy: 0.9578
 - val_kappa:  0.9102

                Validation kappa did not improved from 0.9142642803558032
Epoch 21/30
389/389 [==============================] - 65s 167ms/step - loss: 0.0961 - accuracy: 0.9618 - val_loss: 0.1195 - val_accuracy: 0.9542
 - val_kappa:  0.9077

                Validation kappa did not improved from 0.9142642803558032
Epoch 22/30
389/389 [==============================] - 67s 173ms/step - loss: 0.1050 - accuracy: 0.9614 - val_loss: 0.1127 - val_accuracy: 0.9560
 - val_kappa:  0.9078

                Validation kappa did not improved from 0.9142642803558032
Epoch 23/30
389/389 [==============================] - 67s 171ms/step - loss: 0.0948 - accuracy: 0.9625 - val_loss: 0.1217 - val_accuracy: 0.9585
 - val_kappa:  0.9055

                Validation kappa did not improved from 0.9142642803558032
Epoch 24/30
389/389 [==============================] - 67s 172ms/step - loss: 0.0943 - accuracy: 0.9651 - val_loss: 0.1178 - val_accuracy: 0.9593
 - val_kappa:  0.9144

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b0.h5...
Epoch 25/30
389/389 [==============================] - 67s 173ms/step - loss: 0.0907 - accuracy: 0.9644 - val_loss: 0.1147 - val_accuracy: 0.9593
 - val_kappa:  0.9118

                Validation kappa did not improved from 0.914447319965079
Epoch 26/30
389/389 [==============================] - 67s 172ms/step - loss: 0.0871 - accuracy: 0.9658 - val_loss: 0.1100 - val_accuracy: 0.9600
 - val_kappa:  0.9105

                Validation kappa did not improved from 0.914447319965079
Epoch 27/30
389/389 [==============================] - 67s 173ms/step - loss: 0.0787 - accuracy: 0.9692 - val_loss: 0.1302 - val_accuracy: 0.9535
 - val_kappa:  0.8986

                Validation kappa did not improved from 0.914447319965079
Epoch 28/30
389/389 [==============================] - 67s 171ms/step - loss: 0.0841 - accuracy: 0.9678 - val_loss: 0.1258 - val_accuracy: 0.9556
 - val_kappa:  0.9038

                Validation kappa did not improved from 0.914447319965079
Epoch 29/30
389/389 [==============================] - 67s 172ms/step - loss: 0.0783 - accuracy: 0.9695 - val_loss: 0.1235 - val_accuracy: 0.9575
 - val_kappa:  0.9068

                Validation kappa did not improved from 0.914447319965079
Epoch 30/30
389/389 [==============================] - 67s 172ms/step - loss: 0.0864 - accuracy: 0.9679 - val_loss: 0.1102 - val_accuracy: 0.9611
 - val_kappa:  0.9135
```

```
Validation kappa did not improved from 0.914447319965079
```

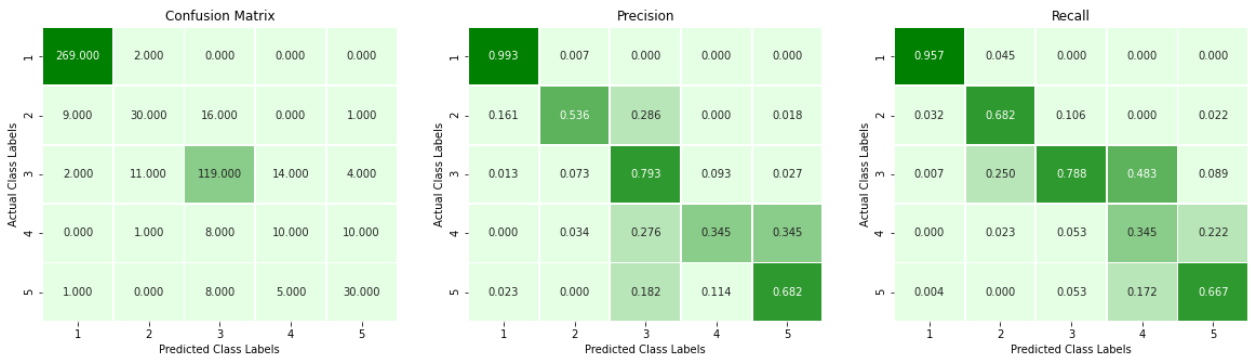
In [ ]:
```
efficientnet_ = efficientnet_b0()
efficientnet_.load_weights("/content/drive/My Drive/models/efficientnet_b0.h5")
result1 = efficientnet_.evaluate(x_validation,labels_validation, verbose = 2)
y_pred = efficientnet_.predict(x_validation, batch_size = 8)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result1[0],4),np.round(result1[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
After running the model for 30 epochs we got loss = 0.1178 Accuracy = 0.9593 kappa_score = 0.9144 on validation data
```

In [ ]:
```
ytrain_efficientb0 = efficientnet_.predict(x_train)
ytrain_efficientb0 = test_prediction(ytrain_efficientb0)
print("First five data points predictions in training:",ytrain_efficientb0[:5])
print("length of traindata prediction:",ytrain_efficientb0.shape,"\n")

yvalidation_efficientb0 = efficientnet_.predict(x_validation)
yvalidation_efficientb0 = test_prediction(yvalidation_efficientb0)
print("First five data points predictions in validation:",yvalidation_efficientb0[:5])
print("length of validation data prediction:",yvalidation_efficientb0.shape,"\n")

ytest_efficientb0 = efficientnet_.predict(x_test)
ytest_efficientb0 = test_prediction(ytest_efficientb0)
print("First five data points predictions in test:",ytest_efficientb0[:5])
print("length of test data prediction:",ytest_efficientb0.shape)
```

```
First five data points predictions in training: [0 1 0 2 3]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 1 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 3 2 2 2]
length of test data prediction: (1928,)
```

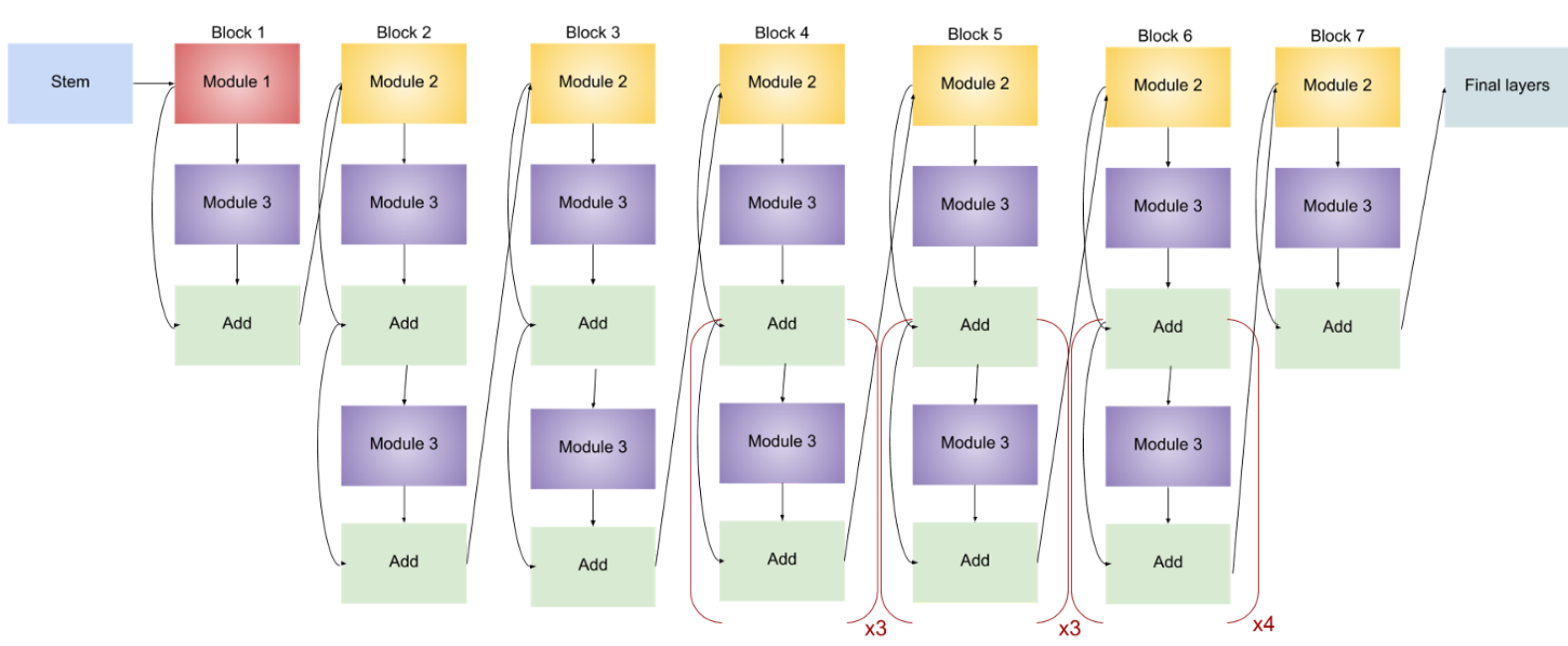
In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/efficientnet_b0'
```

Tensorboard Visualization

EfficientNet-B3 Architecture:



In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def efficientnet_b3():
    '''This function is used for building a model architecture of pretrained EfficientB3 on imagenet data set.'''
    efficientnet_ = EfficientNetB3(weights = 'imagenet',include_top = False, input_shape = (512,512,3) )
    x = global_average_pooling_layer(efficientnet_.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(efficientnet_.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
efficientnet_ = efficientnet_b3()
efficientnet_.summary()
```

```
Model: "Efficientnet-B3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnet-b3 (Functional) (None, 16, 16, 1536)      10783528
_________________________________________________________________
global_average_pooling2d_15  (None, 1536)              0
_________________________________________________________________
dropout_19 (Dropout)         (None, 1536)              0
_________________________________________________________________
dense_29 (Dense)             (None, 5)                 7685
=================================================================
Total params: 10,791,213
Trainable params: 10,703,917
Non-trainable params: 87,296
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/efficientnet_b3')
kappa_metrics = Metrics('/content/drive/My Drive/models/efficientnet_b3.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
efficientnet_ = efficientnet_b3()
history = efficientnet_.fit_generator(
                   data_generator.flow(x_train, labels_train, batch_size = 8),
                    steps_per_epoch=len(x_train) / 8,
                    epochs=30,
                    initial_epoch=0,
                    verbose=1,
                    validation_data=(x_validation, labels_validation),
                    validation_steps=len(x_validation) / 8,
                    callbacks=[kappa_metrics,tensorboard], class_weight = class_weights)
```

```
Downloading data from https://github.com/Callidior/keras-applications/releases/download/efficientnet/efficientnet-b3_weights_tf_dim_ordering_tf_kernels_autoaugment_notop.h5 (https://github.com/Callidior/keras-applications/releases/download/efficientnet/efficientnet-b3_weights_tf_dim_ordering_tf_kernels_autoaugment_notop.h5)
44113920/44107200 [==============================] - 2s 0us/step
Epoch 1/30
  2/389 [..............................] - ETA: 1:38:05 - loss: 0.8306 - accuracy: 0.3250

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (2.143740). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 143s 367ms/step - loss: 0.3380 - accuracy: 0.8572 - val_loss: 0.1959 - val_accuracy: 0.9200
 - val_kappa:  0.7555

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 2/30
389/389 [==============================] - 107s 275ms/step - loss: 0.1958 - accuracy: 0.9227 - val_loss: 0.1531 - val_accuracy: 0.9396
 - val_kappa:  0.8484

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 3/30
389/389 [==============================] - 107s 274ms/step - loss: 0.1682 - accuracy: 0.9329 - val_loss: 0.1491 - val_accuracy: 0.9436
 - val_kappa:  0.8543

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 4/30
389/389 [==============================] - 108s 276ms/step - loss: 0.1563 - accuracy: 0.9381 - val_loss: 0.1317 - val_accuracy: 0.9487
 - val_kappa:  0.8728

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 5/30
389/389 [==============================] - 107s 275ms/step - loss: 0.1502 - accuracy: 0.9403 - val_loss: 0.1262 - val_accuracy: 0.9516
 - val_kappa:  0.8964

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 6/30
389/389 [==============================] - 107s 276ms/step - loss: 0.1440 - accuracy: 0.9463 - val_loss: 0.1229 - val_accuracy: 0.9527
 - val_kappa:  0.8877

                Validation kappa did not improved from 0.8964354671911147
Epoch 7/30
389/389 [==============================] - 107s 275ms/step - loss: 0.1268 - accuracy: 0.9512 - val_loss: 0.1231 - val_accuracy: 0.9545
 - val_kappa:  0.8882

                Validation kappa did not improved from 0.8964354671911147
Epoch 8/30
389/389 [==============================] - 106s 274ms/step - loss: 0.1249 - accuracy: 0.9517 - val_loss: 0.1193 - val_accuracy: 0.9578
 - val_kappa:  0.9076

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 9/30
389/389 [==============================] - 107s 275ms/step - loss: 0.1226 - accuracy: 0.9524 - val_loss: 0.1192 - val_accuracy: 0.9585
 - val_kappa:  0.9067

                Validation kappa did not improved from 0.9075675942859968
Epoch 10/30
389/389 [==============================] - 107s 275ms/step - loss: 0.1195 - accuracy: 0.9526 - val_loss: 0.1108 - val_accuracy: 0.9571
 - val_kappa:  0.9146

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 11/30
389/389 [==============================] - 107s 275ms/step - loss: 0.1161 - accuracy: 0.9553 - val_loss: 0.1165 - val_accuracy: 0.9556
 - val_kappa:  0.9040

                Validation kappa did not improved from 0.9145799871427645
Epoch 12/30
389/389 [==============================] - 107s 276ms/step - loss: 0.1100 - accuracy: 0.9571 - val_loss: 0.1198 - val_accuracy: 0.9553
 - val_kappa:  0.9004

                Validation kappa did not improved from 0.9145799871427645
Epoch 13/30
389/389 [==============================] - 108s 277ms/step - loss: 0.1055 - accuracy: 0.9584 - val_loss: 0.1093 - val_accuracy: 0.9589
 - val_kappa:  0.9080

                Validation kappa did not improved from 0.9145799871427645
Epoch 14/30
389/389 [==============================] - 107s 276ms/step - loss: 0.1077 - accuracy: 0.9585 - val_loss: 0.1144 - val_accuracy: 0.9560
 - val_kappa:  0.9104

                Validation kappa did not improved from 0.9145799871427645
Epoch 15/30
389/389 [==============================] - 108s 277ms/step - loss: 0.1077 - accuracy: 0.9598 - val_loss: 0.1155 - val_accuracy: 0.9535
 - val_kappa:  0.9023

                Validation kappa did not improved from 0.9145799871427645
Epoch 16/30
389/389 [==============================] - 107s 276ms/step - loss: 0.0969 - accuracy: 0.9617 - val_loss: 0.1070 - val_accuracy: 0.9589
 - val_kappa:  0.9148

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 17/30
389/389 [==============================] - 107s 275ms/step - loss: 0.0966 - accuracy: 0.9611 - val_loss: 0.1164 - val_accuracy: 0.9575
 - val_kappa:  0.9109

                Validation kappa did not improved from 0.9147601892621445
Epoch 18/30
389/389 [==============================] - 107s 275ms/step - loss: 0.0957 - accuracy: 0.9620 - val_loss: 0.1211 - val_accuracy: 0.9509
 - val_kappa:  0.9001

                Validation kappa did not improved from 0.9147601892621445
Epoch 19/30
389/389 [==============================] - 107s 275ms/step - loss: 0.0953 - accuracy: 0.9630 - val_loss: 0.1125 - val_accuracy: 0.9567
 - val_kappa:  0.9121

                Validation kappa did not improved from 0.9147601892621445
Epoch 20/30
389/389 [==============================] - 106s 273ms/step - loss: 0.0936 - accuracy: 0.9646 - val_loss: 0.1189 - val_accuracy: 0.9553
 - val_kappa:  0.9085

                Validation kappa did not improved from 0.9147601892621445
Epoch 21/30
389/389 [==============================] - 107s 275ms/step - loss: 0.0814 - accuracy: 0.9692 - val_loss: 0.1093 - val_accuracy: 0.9604
 - val_kappa:  0.9128

                Validation kappa did not improved from 0.9147601892621445
Epoch 22/30
389/389 [==============================] - 108s 278ms/step - loss: 0.0793 - accuracy: 0.9691 - val_loss: 0.1048 - val_accuracy: 0.9615
 - val_kappa:  0.9199

                Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b3.h5...
Epoch 23/30
389/389 [==============================] - 109s 279ms/step - loss: 0.0855 - accuracy: 0.9659 - val_loss: 0.1101 - val_accuracy: 0.9589
 - val_kappa:  0.9182

                Validation kappa did not improved from 0.919873603430764
Epoch 24/30
389/389 [==============================] - 107s 276ms/step - loss: 0.0770 - accuracy: 0.9695 - val_loss: 0.1151 - val_accuracy: 0.9556
 - val_kappa:  0.9024

                Validation kappa did not improved from 0.919873603430764
Epoch 25/30
389/389 [==============================] - 107s 275ms/step - loss: 0.0789 - accuracy: 0.9692 - val_loss: 0.1168 - val_accuracy: 0.9527
 - val_kappa:  0.9073

                Validation kappa did not improved from 0.919873603430764
Epoch 26/30
389/389 [==============================] - 106s 273ms/step - loss: 0.0760 - accuracy: 0.9693 - val_loss: 0.1117 - val_accuracy: 0.9600
 - val_kappa:  0.9059

                Validation kappa did not improved from 0.919873603430764
Epoch 27/30
389/389 [==============================] - 105s 270ms/step - loss: 0.0689 - accuracy: 0.9719 - val_loss: 0.1138 - val_accuracy: 0.9571
 - val_kappa:  0.9057

                Validation kappa did not improved from 0.919873603430764
Epoch 28/30
389/389 [==============================] - 108s 277ms/step - loss: 0.0750 - accuracy: 0.9701 - val_loss: 0.1184 - val_accuracy: 0.9567
 - val_kappa:  0.8994

                Validation kappa did not improved from 0.919873603430764
Epoch 29/30
389/389 [==============================] - 107s 275ms/step - loss: 0.0683 - accuracy: 0.9728 - val_loss: 0.1210 - val_accuracy: 0.9575
 - val_kappa:  0.9078

                Validation kappa did not improved from 0.919873603430764
```

```
Epoch 30/30
389/389 [==============================] - 107s 276ms/step - loss: 0.0659 - accuracy: 0.9742 - val_loss: 0.1220 - val_accuracy: 0.9564
 - val_kappa:  0.9046

                Validation kappa did not improved from 0.919873603430764
```

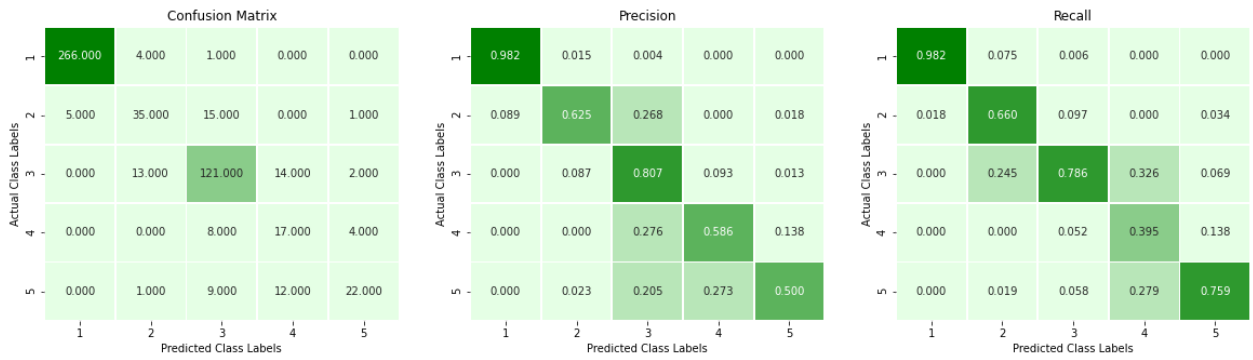
In [ ]:
```
efficientnet_ = efficientnet_b3()
efficientnet_.load_weights("/content/drive/My Drive/models/efficientnet_b3.h5")
result1 = efficientnet_.evaluate(x_validation,labels_validation, verbose = 2)
y_pred = efficientnet_.predict(x_validation, batch_size = 8)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result1[0],4),np.round(result1[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
After running the model for 30 epochs we got loss = 0.1048 Accuracy = 0.9615 kappa_score = 0.9199 on validation data
```

In [ ]:
```
ytrain_efficientb1 = efficientnet_.predict(x_train)
ytrain_efficientb1 = test_prediction(ytrain_efficientb1)
print("First five data points predictions in training:",ytrain_efficientb1[:5])
print("length of traindata prediction:",ytrain_efficientb1.shape,"\n")

yvalidation_efficientb1 = efficientnet_.predict(x_validation)
yvalidation_efficientb1 = test_prediction(yvalidation_efficientb1)
print("First five data points predictions in validation:",yvalidation_efficientb1[:5])
print("length of validation data prediction:",yvalidation_efficientb1.shape,"\n")

ytest_efficientb1 = efficientnet_.predict(x_test)
ytest_efficientb1 = test_prediction(ytest_efficientb1)
print("First five data points predictions in test:",ytest_efficientb1[:5])
print("length of test data prediction:",ytest_efficientb1.shape)
```

```
First five data points predictions in training: [0 1 0 2 3]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 1 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 3 2 2 2]
length of test data prediction: (1928,)
```

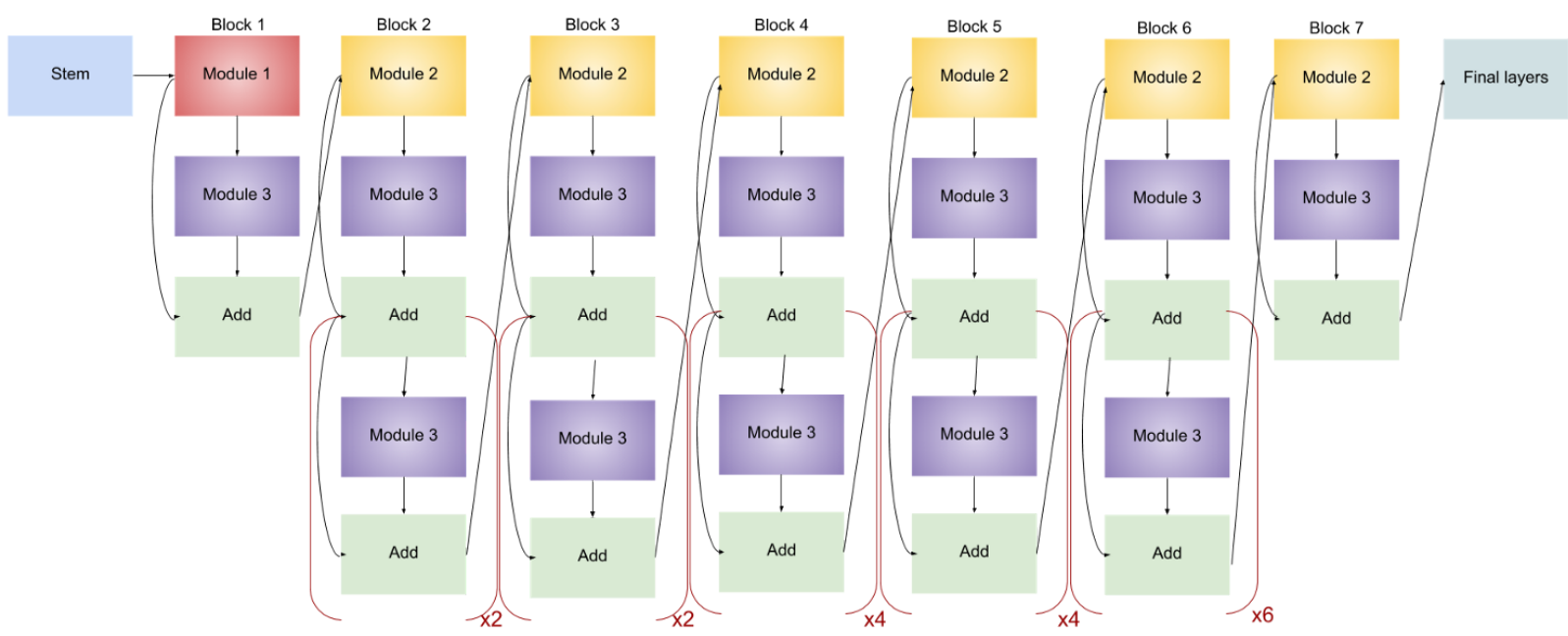
In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/efficientnet_b3'
```

Tensorboard Visualization:

EfficientNet-B4:



In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def efficientnet_b4():
    '''This function is used for building a model architecture of pretrained EfficientB4 on imagenet data set.'''
    efficientnet_ = EfficientNetB4(weights = 'imagenet',include_top = False, input_shape = (512,512,3) )
    x = global_average_pooling_layer(efficientnet_.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(efficientnet_.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
efficientnet_ = efficientnet_b4()
efficientnet_.summary()
```

```
Model: "Efficientnet-B4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
efficientnet-b4 (Functional) (None, 16, 16, 1792)      17673816
_________________________________________________________________
global_average_pooling2d_17  (None, 1792)              0
_________________________________________________________________
dropout_21 (Dropout)         (None, 1792)              0
_________________________________________________________________
dense_31 (Dense)             (None, 5)                 8965
=================================================================
Total params: 17,682,781
Trainable params: 17,557,581
Non-trainable params: 125,200
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/efficientnet_b4')
kappa_metrics = Metrics('/content/drive/My Drive/models/efficientnet_b4.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
efficientnet_ = efficientnet_b4()
history = efficientnet_.fit_generator(
                   data_generator.flow(x_train, labels_train, batch_size = 8),
                    steps_per_epoch=len(x_train) / 8,
                    epochs=30,
                    initial_epoch=0,
                    verbose=1,
                    validation_data=(x_validation, labels_validation),
                    validation_steps=len(x_validation) / 8,
                    callbacks=[kappa_metrics,tensorboard], class_weight = class_weights)
```

```
Epoch 1/30
  2/389 [..............................] - ETA: 2:16:54 - loss: 0.7094 - accuracy: 0.4750

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (0.670378). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 175s 449ms/step - loss: 0.3441 - accuracy: 0.8629 - val_loss: 0.1866 - val_accuracy: 0.9244
 - val_kappa:  0.7598

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 2/30
389/389 [==============================] - 128s 329ms/step - loss: 0.1906 - accuracy: 0.9264 - val_loss: 0.1494 - val_accuracy: 0.9404
 - val_kappa:  0.8374

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 3/30
389/389 [==============================] - 129s 331ms/step - loss: 0.1707 - accuracy: 0.9334 - val_loss: 0.1347 - val_accuracy: 0.9440
 - val_kappa:  0.8634

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 4/30
389/389 [==============================] - 129s 331ms/step - loss: 0.1527 - accuracy: 0.9400 - val_loss: 0.1338 - val_accuracy: 0.9480
 - val_kappa:  0.8702

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 5/30
389/389 [==============================] - 129s 332ms/step - loss: 0.1402 - accuracy: 0.9432 - val_loss: 0.1391 - val_accuracy: 0.9480
 - val_kappa:  0.8652

               Validation kappa did not improved from 0.8702038350440205
Epoch 6/30
389/389 [==============================] - 128s 330ms/step - loss: 0.1320 - accuracy: 0.9479 - val_loss: 0.1272 - val_accuracy: 0.9498
 - val_kappa:  0.8915

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 7/30
389/389 [==============================] - 129s 332ms/step - loss: 0.1305 - accuracy: 0.9474 - val_loss: 0.1182 - val_accuracy: 0.9564
 - val_kappa:  0.9013

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 8/30
389/389 [==============================] - 129s 332ms/step - loss: 0.1248 - accuracy: 0.9513 - val_loss: 0.1160 - val_accuracy: 0.9556
 - val_kappa:  0.8969

               Validation kappa did not improved from 0.9013315032030756
Epoch 9/30
389/389 [==============================] - 129s 331ms/step - loss: 0.1146 - accuracy: 0.9558 - val_loss: 0.1252 - val_accuracy: 0.9535
 - val_kappa:  0.8872

               Validation kappa did not improved from 0.9013315032030756
Epoch 10/30
389/389 [==============================] - 129s 330ms/step - loss: 0.1170 - accuracy: 0.9547 - val_loss: 0.1165 - val_accuracy: 0.9575
 - val_kappa:  0.9055

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 11/30
389/389 [==============================] - 129s 331ms/step - loss: 0.1132 - accuracy: 0.9562 - val_loss: 0.1122 - val_accuracy: 0.9582
 - val_kappa:  0.9084

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 12/30
389/389 [==============================] - 128s 330ms/step - loss: 0.1047 - accuracy: 0.9575 - val_loss: 0.1161 - val_accuracy: 0.9560
 - val_kappa:  0.9036

               Validation kappa did not improved from 0.9084427092272792
Epoch 13/30
389/389 [==============================] - 128s 328ms/step - loss: 0.1079 - accuracy: 0.9582 - val_loss: 0.1220 - val_accuracy: 0.9527
 - val_kappa:  0.8898

               Validation kappa did not improved from 0.9084427092272792
Epoch 14/30
389/389 [==============================] - 128s 329ms/step - loss: 0.1038 - accuracy: 0.9594 - val_loss: 0.1151 - val_accuracy: 0.9582
 - val_kappa:  0.9127

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 15/30
389/389 [==============================] - 128s 330ms/step - loss: 0.0953 - accuracy: 0.9625 - val_loss: 0.1330 - val_accuracy: 0.9495
 - val_kappa:  0.8837

               Validation kappa did not improved from 0.9126502876136571
Epoch 16/30
389/389 [==============================] - 130s 333ms/step - loss: 0.0946 - accuracy: 0.9641 - val_loss: 0.1092 - val_accuracy: 0.9571
 - val_kappa:  0.9079

               Validation kappa did not improved from 0.9126502876136571
Epoch 17/30
389/389 [==============================] - 128s 330ms/step - loss: 0.0937 - accuracy: 0.9636 - val_loss: 0.1157 - val_accuracy: 0.9571
 - val_kappa:  0.9078

               Validation kappa did not improved from 0.9126502876136571
Epoch 18/30
389/389 [==============================] - 129s 331ms/step - loss: 0.0864 - accuracy: 0.9671 - val_loss: 0.1061 - val_accuracy: 0.9578
 - val_kappa:  0.9136

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 19/30
389/389 [==============================] - 128s 330ms/step - loss: 0.0886 - accuracy: 0.9669 - val_loss: 0.1146 - val_accuracy: 0.9578
 - val_kappa:  0.9152

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 20/30
389/389 [==============================] - 128s 329ms/step - loss: 0.0907 - accuracy: 0.9662 - val_loss: 0.1119 - val_accuracy: 0.9611
 - val_kappa:  0.9182

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 21/30
389/389 [==============================] - 128s 329ms/step - loss: 0.0794 - accuracy: 0.9687 - val_loss: 0.1267 - val_accuracy: 0.9564
 - val_kappa:  0.9075

               Validation kappa did not improved from 0.9182403420455993
Epoch 22/30
389/389 [==============================] - 128s 330ms/step - loss: 0.0795 - accuracy: 0.9690 - val_loss: 0.1138 - val_accuracy: 0.9593
 - val_kappa:  0.9185

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 23/30
389/389 [==============================] - 129s 331ms/step - loss: 0.0818 - accuracy: 0.9709 - val_loss: 0.1198 - val_accuracy: 0.9600
 - val_kappa:  0.9193

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/efficientnet_b4.h5...
Epoch 24/30
389/389 [==============================] - 128s 330ms/step - loss: 0.0742 - accuracy: 0.9706 - val_loss: 0.1137 - val_accuracy: 0.9582
 - val_kappa:  0.9113

               Validation kappa did not improved from 0.9192922172017634
Epoch 25/30
389/389 [==============================] - 128s 330ms/step - loss: 0.0714 - accuracy: 0.9722 - val_loss: 0.1117 - val_accuracy: 0.9564
 - val_kappa:  0.9095

               Validation kappa did not improved from 0.9192922172017634
Epoch 26/30
389/389 [==============================] - 130s 334ms/step - loss: 0.0698 - accuracy: 0.9728 - val_loss: 0.1168 - val_accuracy: 0.9582
 - val_kappa:  0.9093

               Validation kappa did not improved from 0.9192922172017634
Epoch 27/30
389/389 [==============================] - 132s 338ms/step - loss: 0.0657 - accuracy: 0.9750 - val_loss: 0.1272 - val_accuracy: 0.9498
 - val_kappa:  0.8924

               Validation kappa did not improved from 0.9192922172017634
Epoch 28/30
389/389 [==============================] - 132s 338ms/step - loss: 0.0655 - accuracy: 0.9742 - val_loss: 0.1235 - val_accuracy: 0.9538
 - val_kappa:  0.9081

               Validation kappa did not improved from 0.9192922172017634
Epoch 29/30
389/389 [==============================] - 132s 339ms/step - loss: 0.0606 - accuracy: 0.9763 - val_loss: 0.1208 - val_accuracy: 0.9553
 - val_kappa:  0.9090

               Validation kappa did not improved from 0.9192922172017634
Epoch 30/30
389/389 [==============================] - 131s 338ms/step - loss: 0.0614 - accuracy: 0.9767 - val_loss: 0.1170 - val_accuracy: 0.9600
 - val_kappa:  0.9136
```

```
Validation kappa did not improved from 0.9192922172017634
```

In [ ]:
```
efficientnet_ = efficientnet_b4()
efficientnet_.load_weights("/content/drive/My Drive/models/efficientnet_b4.h5")
result1 = efficientnet_.evaluate(x_validation,labels_validation, verbose = 2)
y_pred = efficientnet_.predict(x_validation, batch_size = 8)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result1[0],4),np.round(result1[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

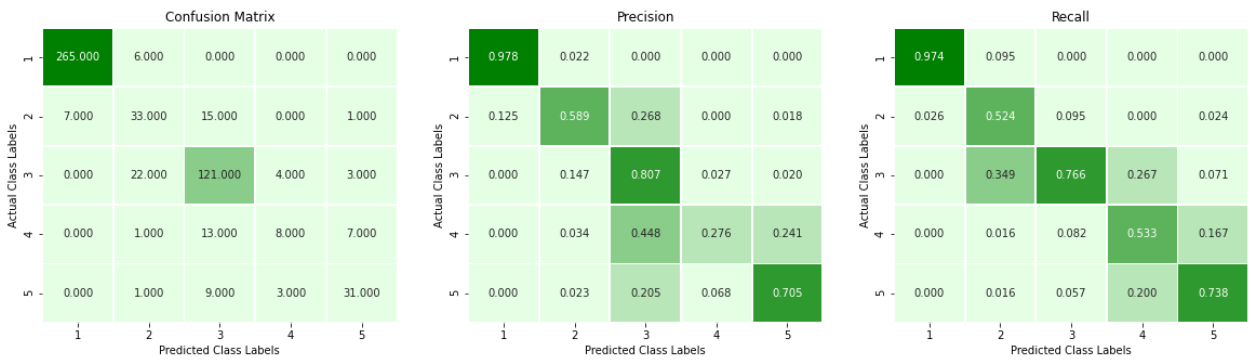
```
After running the model for 30 epochs we got loss = 0.1198 Accuracy = 0.96 kappa_score = 0.9193 on validation data
```

In [ ]:
```
ytrain_efficientb2 = efficientnet_.predict(x_train)
ytrain_efficientb2 = test_prediction(ytrain_efficientb2)
print("First five data points predictions in training:",ytrain_efficientb2[:5])
print("length of traindata prediction:",ytrain_efficientb2.shape,"\n")

yvalidation_efficientb2 = efficientnet_.predict(x_validation)
yvalidation_efficientb2 = test_prediction(yvalidation_efficientb2)
print("First five data points predictions in validation:",yvalidation_efficientb2[:5])
print("length of validation data prediction:",yvalidation_efficientb2.shape,"\n")

ytest_efficientb2 = efficientnet_.predict(x_test)
ytest_efficientb2 = test_prediction(ytest_efficientb2)
print("First five data points predictions in test:",ytest_efficientb2[:5])
print("length of test data prediction:",ytest_efficientb2.shape)
```

```
First five data points predictions in training: [0 1 0 2 2]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 1 4]
length of validation data prediction: (550,)

First five data points predictions in test: [1 3 3 2 3]
length of test data prediction: (1928,)
```

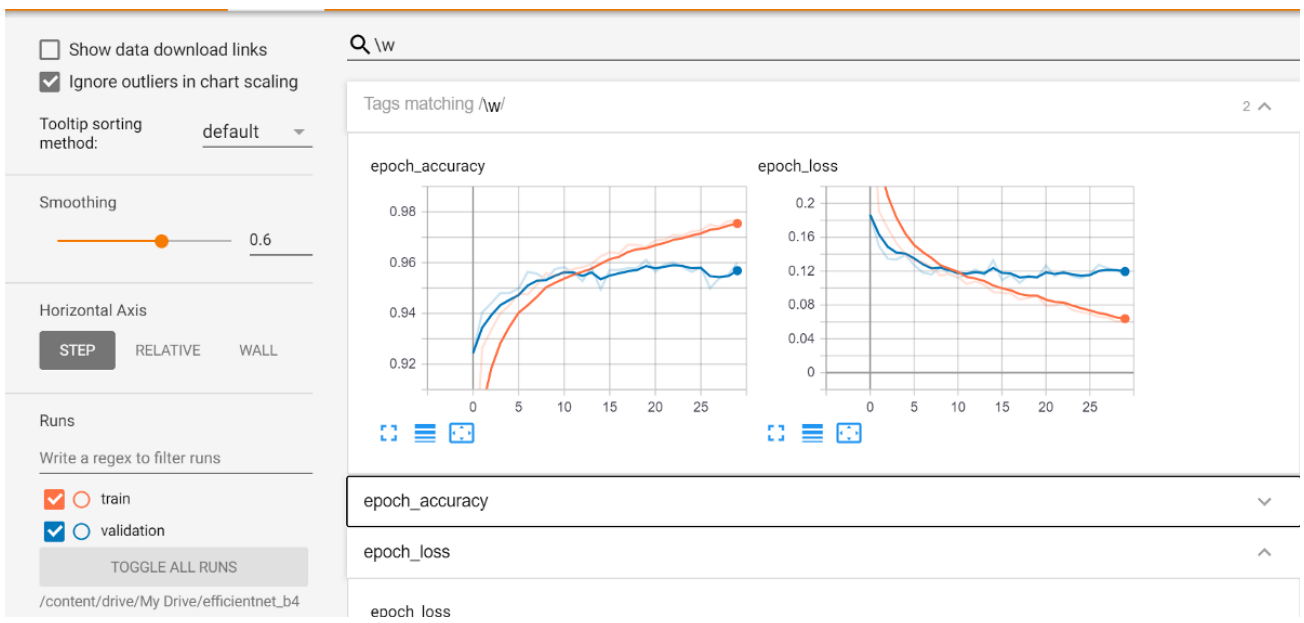
In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```

```
/usr/local/lib/python3.6/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the functions in the public API at pandas.testing instead.
  import pandas.util.testing as tm
```

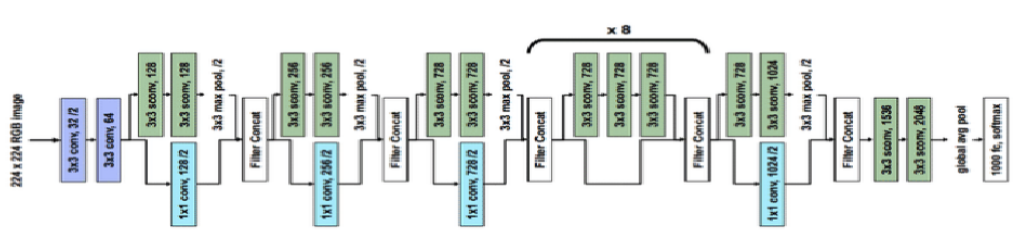


In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/efficientnet_b4'
```

Tensorboard Visualization:



Xception Architecture:



XCeption is an efficient architecture that relies on two main points :

- Depthwise Separable Convolution:
  To overcome the cost of such operations, depthwise separable convolutions have been introduced.
  They are themselves divided into 2 main steps :
  1. Depthwise Convolution
  2. Pointwise Convolution
- Shortcuts between Convolution blocks as in ResNet:
  These are skip connections which are introduced in Resnet.

Read more about Xception net here (https://maelfabien.github.io/deeplearning/xception/#ii-in-keras).

In [ ]:
```
global_average_pooling_layer = GAP2D()
dropout_layer = dropout()
dense_layer = dense()
```

In [ ]:
```
def xception():
    '''This function is used for building a model architecture of pretrained Xception on imagenet data set.'''
    xception_ = Xception(weights = 'imagenet',include_top = False, input_shape = (512,512,3) )
    x = global_average_pooling_layer(xception_.layers[-1].output)
    x = dropout_layer(x)
    output = dense_layer(x)
    model = Model(xception_.layers[0].input,output)
    model.compile(loss='binary_crossentropy', optimizer=keras.optimizers.Adam(lr=0.00005), metrics=['accuracy'])
    return model
```

In [ ]:
```
xception_ = xception()
xception_.summary()
```

```
Model: "Xception"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
xception (Functional)        (None, 16, 16, 2048)      20861480
_________________________________________________________________
global_average_pooling2d_19  (None, 2048)              0
_________________________________________________________________
dropout_23 (Dropout)         (None, 2048)              0
_________________________________________________________________
dense_33 (Dense)             (None, 5)                 10245
=================================================================
Total params: 20,871,725
Trainable params: 20,817,197
Non-trainable params: 54,528
_________________________________________________________________
```

In [ ]:
```
tensorboard = TensorBoard(log_dir = '/content/drive/My Drive/xception')
kappa_metrics = Metrics('/content/drive/My Drive/models/xception.h5')
data_generator = ImageDataGenerator(horizontal_flip=True,vertical_flip=True,rotation_range=180,zoom_range = 0.2)
xception_ = xception()
history = xception_.fit_generator(
                  data_generator.flow(x_train, labels_train, batch_size = 8),
                   steps_per_epoch=len(x_train) / 8,
                   epochs=30,
                   initial_epoch=0,
                   verbose=1,
                   validation_data=(x_validation, labels_validation),
                   validation_steps=len(x_validation) / 8,
                   callbacks=[kappa_metrics,tensorboard], class_weight = class_weights)
```

```
Downloading data from https://github.com/fchollet/deep-learning-models/releases/download/v0.4/xception_weights_tf_dim_ordering_tf_kernels_notop.h5 (https://github.com/fchollet/deep-learning-models/releases/download/v0.4/xception_weights_tf_dim_ordering_tf_kernels_notop.h5)
83689472/83683744 [==============================] - 3s 0us/step
Epoch 1/30
  3/389 [..............................] - ETA: 28:37 - loss: 0.6218 - accuracy: 0.7167

/usr/local/lib/python3.6/dist-packages/keras/callbacks/callbacks.py:95: RuntimeWarning: Method (on_train_batch_end) is slow compared to the batch update (1.466768). Check your callbacks.
  % (hook_name, delta_t_median), RuntimeWarning)

389/389 [==============================] - 90s 231ms/step - loss: 0.2971 - accuracy: 0.8906 - val_loss: 0.1630 - val_accuracy: 0.9404
 - val_kappa:  0.8357

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 2/30
389/389 [==============================] - 76s 195ms/step - loss: 0.1871 - accuracy: 0.9265 - val_loss: 0.1628 - val_accuracy: 0.9342
 - val_kappa:  0.8622

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 3/30
389/389 [==============================] - 76s 195ms/step - loss: 0.1653 - accuracy: 0.9335 - val_loss: 0.1320 - val_accuracy: 0.9509
 - val_kappa:  0.8756

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 4/30
389/389 [==============================] - 76s 195ms/step - loss: 0.1512 - accuracy: 0.9409 - val_loss: 0.1329 - val_accuracy: 0.9498
 - val_kappa:  0.8734

               Validation kappa did not improved from 0.8755509324296791
Epoch 5/30
389/389 [==============================] - 75s 193ms/step - loss: 0.1404 - accuracy: 0.9443 - val_loss: 0.1368 - val_accuracy: 0.9433
 - val_kappa:  0.8801

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 6/30
389/389 [==============================] - 75s 192ms/step - loss: 0.1342 - accuracy: 0.9483 - val_loss: 0.1208 - val_accuracy: 0.9520
 - val_kappa:  0.8927

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 7/30
389/389 [==============================] - 74s 191ms/step - loss: 0.1295 - accuracy: 0.9499 - val_loss: 0.1150 - val_accuracy: 0.9571
 - val_kappa:  0.8994

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 8/30
389/389 [==============================] - 74s 191ms/step - loss: 0.1229 - accuracy: 0.9533 - val_loss: 0.1099 - val_accuracy: 0.9571
 - val_kappa:  0.9032

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 9/30
389/389 [==============================] - 74s 191ms/step - loss: 0.1128 - accuracy: 0.9557 - val_loss: 0.1244 - val_accuracy: 0.9473
 - val_kappa:  0.8871

               Validation kappa did not improved from 0.903238170237988
Epoch 10/30
389/389 [==============================] - 74s 191ms/step - loss: 0.1124 - accuracy: 0.9576 - val_loss: 0.1090 - val_accuracy: 0.9589
 - val_kappa:  0.9066

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 11/30
389/389 [==============================] - 75s 193ms/step - loss: 0.1115 - accuracy: 0.9567 - val_loss: 0.1227 - val_accuracy: 0.9429
 - val_kappa:  0.8766

               Validation kappa did not improved from 0.9065610506019074
Epoch 12/30
389/389 [==============================] - 74s 190ms/step - loss: 0.1049 - accuracy: 0.9598 - val_loss: 0.1051 - val_accuracy: 0.9578
 - val_kappa:  0.9064

               Validation kappa did not improved from 0.9065610506019074
Epoch 13/30
389/389 [==============================] - 74s 190ms/step - loss: 0.0991 - accuracy: 0.9612 - val_loss: 0.1114 - val_accuracy: 0.9531
 - val_kappa:  0.8982

               Validation kappa did not improved from 0.9065610506019074
Epoch 14/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0963 - accuracy: 0.9643 - val_loss: 0.1090 - val_accuracy: 0.9578
 - val_kappa:  0.9068

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 15/30
389/389 [==============================] - 74s 190ms/step - loss: 0.0864 - accuracy: 0.9668 - val_loss: 0.1145 - val_accuracy: 0.9524
 - val_kappa:  0.8969

               Validation kappa did not improved from 0.9068080924463723
Epoch 16/30
389/389 [==============================] - 74s 190ms/step - loss: 0.0858 - accuracy: 0.9679 - val_loss: 0.1161 - val_accuracy: 0.9502
 - val_kappa:  0.8988

               Validation kappa did not improved from 0.9068080924463723
Epoch 17/30
389/389 [==============================] - 74s 190ms/step - loss: 0.0830 - accuracy: 0.9676 - val_loss: 0.1071 - val_accuracy: 0.9575
 - val_kappa:  0.9161

               Validation Kappa has improved. Saving model to /content/drive/My Drive/models/xception.h5...
Epoch 18/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0889 - accuracy: 0.9676 - val_loss: 0.1209 - val_accuracy: 0.9516
 - val_kappa:  0.9040

               Validation kappa did not improved from 0.9161227353138535
Epoch 19/30
389/389 [==============================] - 73s 188ms/step - loss: 0.0743 - accuracy: 0.9717 - val_loss: 0.1168 - val_accuracy: 0.9531
 - val_kappa:  0.9099

               Validation kappa did not improved from 0.9161227353138535
Epoch 20/30
389/389 [==============================] - 73s 188ms/step - loss: 0.0762 - accuracy: 0.9707 - val_loss: 0.1158 - val_accuracy: 0.9538
 - val_kappa:  0.8944

               Validation kappa did not improved from 0.9161227353138535
Epoch 21/30
389/389 [==============================] - 73s 188ms/step - loss: 0.0724 - accuracy: 0.9720 - val_loss: 0.1235 - val_accuracy: 0.9484
 - val_kappa:  0.8913

               Validation kappa did not improved from 0.9161227353138535
Epoch 22/30
389/389 [==============================] - 73s 189ms/step - loss: 0.0664 - accuracy: 0.9749 - val_loss: 0.1230 - val_accuracy: 0.9527
 - val_kappa:  0.8972

               Validation kappa did not improved from 0.9161227353138535
Epoch 23/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0642 - accuracy: 0.9733 - val_loss: 0.1222 - val_accuracy: 0.9542
 - val_kappa:  0.9025

               Validation kappa did not improved from 0.9161227353138535
Epoch 24/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0649 - accuracy: 0.9751 - val_loss: 0.1306 - val_accuracy: 0.9505
 - val_kappa:  0.8916

               Validation kappa did not improved from 0.9161227353138535
Epoch 25/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0649 - accuracy: 0.9755 - val_loss: 0.1285 - val_accuracy: 0.9564
 - val_kappa:  0.9107

               Validation kappa did not improved from 0.9161227353138535
Epoch 26/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0571 - accuracy: 0.9784 - val_loss: 0.1315 - val_accuracy: 0.9556
 - val_kappa:  0.9095

               Validation kappa did not improved from 0.9161227353138535
Epoch 27/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0580 - accuracy: 0.9776 - val_loss: 0.1405 - val_accuracy: 0.9524
 - val_kappa:  0.8975

               Validation kappa did not improved from 0.9161227353138535
Epoch 28/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0629 - accuracy: 0.9767 - val_loss: 0.1350 - val_accuracy: 0.9531
 - val_kappa:  0.8999

               Validation kappa did not improved from 0.9161227353138535
Epoch 29/30
389/389 [==============================] - 74s 189ms/step - loss: 0.0523 - accuracy: 0.9796 - val_loss: 0.1357 - val_accuracy: 0.9520
 - val_kappa:  0.8974

               Validation kappa did not improved from 0.9161227353138535
```

```
Epoch 30/30
389/389 [==============================] - 74s 190ms/step - loss: 0.0485 - accuracy: 0.9821 - val_loss: 0.1498 - val_accuracy: 0.9524
 - val_kappa:  0.8885

                Validation kappa did not improved from 0.9161227353138535
```

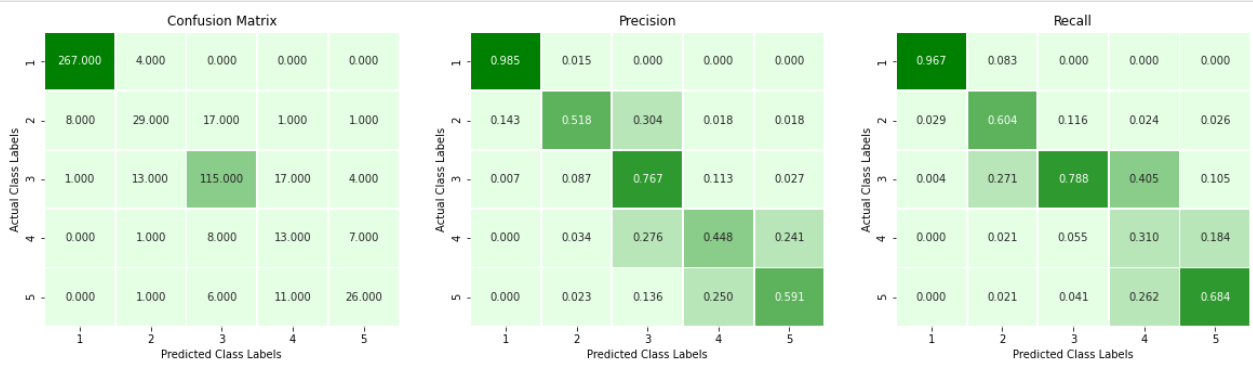
In [ ]:
```
xception_ = xception()
xception_.load_weights("/content/drive/My Drive/models/xception.h5")
result1 = xception_.evaluate(x_validation,labels_validation, verbose = 2)
y_pred = xception_.predict(x_validation, batch_size = 8)
print("After running the model for 30 epochs we got loss = {} Accuracy = {} kappa_score = {} on validation data".format(np.round(result1[0],4),np.round(result1[1],4),np.round(kappa_metric(labels_validation,y_pred),4)))
```

```
After running the model for 30 epochs we got loss = 0.1071 Accuracy = 0.9575 kappa_score = 0.9161 on validation data
```

In [ ]:
```
ytrain_xception = xception_.predict(x_train)
ytrain_xception = test_prediction(ytrain_xception)
print("First five data points predictions in training:",ytrain_xception[:5])
print("length of traindata prediction:",ytrain_xception.shape,"\n")

yvalidation_xception = xception_.predict(x_validation)
yvalidation_xception = test_prediction(yvalidation_xception)
print("First five data points predictions in validation:",yvalidation_xception[:5])
print("length of validation data prediction:",yvalidation_xception.shape,"\n")

ytest_xception = xception_.predict(x_test)
ytest_xception = test_prediction(ytest_xception)
print("First five data points predictions in test:",ytest_xception[:5])
print("length of test data prediction:",ytest_xception.shape)
```

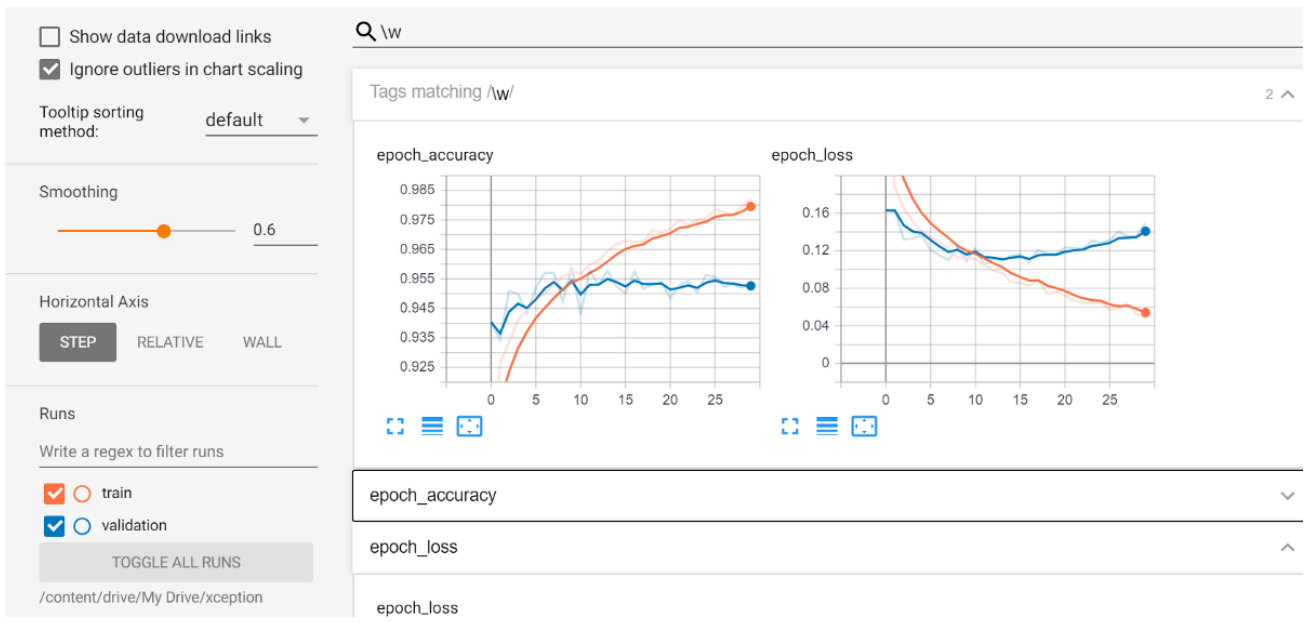
```
First five data points predictions in training: [0 1 0 2 2]
length of traindata prediction: (3112,)

First five data points predictions in validation: [0 0 0 1 4]
length of validation data prediction: (550,)

First five data points predictions in test: [2 3 2 2 2]
length of test data prediction: (1928,)
```

In [ ]:
```
metric = PerformanceMetric(labels_validation, y_pred)
metric.plotting()
```



In [ ]:
```
%reload_ext tensorboard
%tensorboard --logdir='/content/drive/My Drive/xception'
```

Tensorboard Visualization:



Saving Files for further tasks:

In [ ]:
```
train_predictions = pd.DataFrame(ytrain_baseline, columns=['baseline'])
train_predictions['alexnet']=ytrain_alexnet
train_predictions['vgg16']=ytrain_vgg16
train_predictions['vgg19']=ytrain_vgg19
train_predictions['densenet']=ytrain_densenet
train_predictions['resnet']=ytrain_resnet
train_predictions['resnet1']=ytrain_resnet1
train_predictions['inception']=ytrain_inception
train_predictions['efficientb0']=ytrain_efficientb0
train_predictions['efficientb1']=ytrain_efficientb1
train_predictions['efficientb2']=ytrain_efficientb2
train_predictions['xception']=ytrain_xception
train_predictions['label']=train_labels
train_predictions.head(3)
```

Out[120]:

	baseline	alexnet	vgg16	vgg19	densenet	resnet	resnet1	inception	efficientb0	efficientb1	efficientb2	xception	label
0	0	3	0	0	0	0	0	0	0	0	0	0	0
1	2	3	2	2	1	2	1	2	1	1	1	1	1
2	0	3	0	0	0	0	0	0	0	0	0	0	0

In [ ]:
```
validation_predictions = pd.DataFrame(yvalidation_baseline, columns=['baseline'])
validation_predictions['alexnet']=yvalidation_alexnet
validation_predictions['vgg16']=yvalidation_vgg16
validation_predictions['vgg19']=yvalidation_vgg19
validation_predictions['densenet']=yvalidation_densenet
validation_predictions['resnet']=yvalidation_resnet
validation_predictions['resnet1']=yvalidation_resnet1
validation_predictions['inception']=yvalidation_inception
validation_predictions['efficientb0']=yvalidation_efficientb0
validation_predictions['efficientb1']=yvalidation_efficientb1
validation_predictions['efficientb2']=yvalidation_efficientb2
validation_predictions['xception']=yvalidation_xception
validation_predictions['label']=validation_labels
validation_predictions.head(3)
```

Out[122]:

	baseline	alexnet	vgg16	vgg19	densenet	resnet	resnet1	inception	efficientb0	efficientb1	efficientb2	xception	label
0	0	3	0	0	0	0	0	0	0	0	0	0	0
1	0	3	0	0	0	0	0	0	0	0	0	0	0
2	2	3	0	0	0	0	0	0	0	0	0	0	0

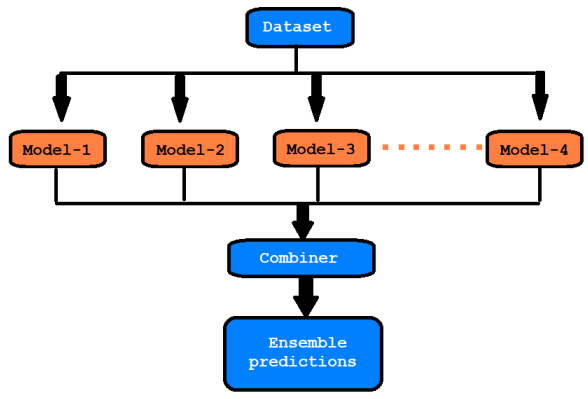
In [ ]:
```
test_predictions = pd.DataFrame(ytest_baseline, columns=['baseline'])
test_predictions['alexnet']=ytest_alexnet
test_predictions['vgg16']=ytest_vgg16
test_predictions['vgg19']=ytest_vgg19
test_predictions['densenet']=ytest_densenet
test_predictions['resnet']=ytest_resnet
test_predictions['resnet1']=ytest_resnet1
test_predictions['inception']=ytest_inception
test_predictions['efficientb0']=ytest_efficientb0
test_predictions['efficientb1']=ytest_efficientb1
test_predictions['efficientb2']=ytest_efficientb2
test_predictions['xception']=ytest_xception
test_predictions.head(3)
```

Out[118]:

	baseline	alexnet	vgg16	vgg19	densenet	resnet	resnet1	inception	efficientb0	efficientb1	efficientb2	xception
0	1	3	1	1	1	1	1	1	1	1	1	2
1	3	3	2	3	3	2	2	2	3	3	3	3
2	2	3	3	2	3	3	2	2	2	2	3	2

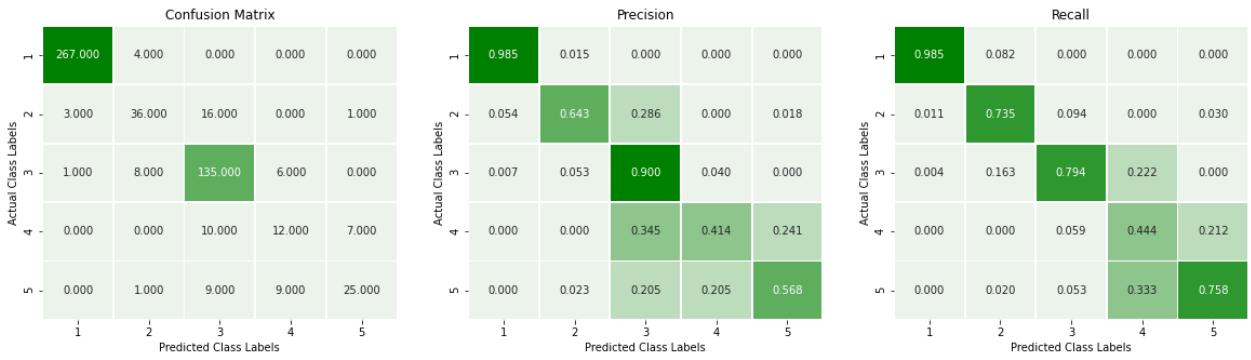
Ensembles:

Taking N model predictions and taking the max count of labels as class predictions.
E.g: [1,0,0,1,1,1,2] will be predicted as 1 since 4 models predicted as 1, 2 models as 0 and 1 model as 2.
Hence the class prediction = max count of labels i.e.,1



In [ ]:
```
def ensembling(df):
    '''
    This process performs ensembling of all pretrained model predictions.
    It takes the max count of each unique label and assigns it as class predictions.
    Arguments:
    df - (pd.dataframe) - dataframe containing predictions of all models.
    '''

    predictions=[]
    for i in range(df.shape[0]):
        row = list(df.iloc[i][2:12])  # ignoring baseline and alexnet predictions
        max=-1; max_label=-1
        for j in set(row):
            count = row.count(j)
            if count>max: max=count; max_label=j   # checking the best agreement label from all models
        predictions.append(max_label)
    return predictions
```

In [ ]:
```
train_preds = ensembling(train_predictions)
val_preds = ensembling(validation_predictions)
test_preds = ensembling(test_predictions)
metric = PerformanceMetric(validation_predictions['label'], val_preds)
metric.plotting()
```



In [ ]:
```
tr_kappa_ = cohen_kappa_score(train_predictions['label'], train_preds,  weights='quadratic' )
val_kappa_ = cohen_kappa_score(validation_predictions['label'], val_preds,  weights='quadratic' )
print("AFter Ensembling of training data we get kappa score of:",tr_kappa_)
print("AFter Ensembling of validation data we get kappa score of:",val_kappa_)
```

```
AFter Ensembling of training data we get kappa score of: 0.9560219551062169
AFter Ensembling of validation data we get kappa score of: 0.9314817445952429
```

In [ ]:
```
train_predictions['ensemble']=train_preds
validation_predictions['ensemble']=val_preds
```

In [ ]:
```
train_predictions.to_csv('/content/drive/My Drive/submission/train.csv',index=False)
validation_predictions.to_csv('/content/drive/My Drive/submission/validation.csv',index=False)
test_predictions.to_csv('/content/drive/My Drive/submission/test.csv',index=False)
```

Overall Results:

In [ ]:
```
table = PrettyTable()
table.field_names = ['Sr.No','Model','val_kappa','Trainable Parameters']
table.add_row(['1','Baseline','0.6931','1,86,037'])
table.add_row(['2','AlexNet','0.7758','512,332,165'])
table.add_row(['3','VGG16','0.8931','9,441,797'])
table.add_row(['4','VGG19','0.9128','16,521,221'])
table.add_row(['5','DenseNet','0.923','6,958,981'])
table.add_row(['6','Resnet50','0.9091','23,544,837'])
table.add_row(['7','Resnet152','0.9146','58,229,765'])
table.add_row(['8','Inceptionv3','0.9014','21,778,597'])
table.add_row(['9','EfficientB0','0.9144','4,013,953'])
table.add_row(['10','EfficientB3','0.9199','10,703,917'])
table.add_row(['11','EfficientB4','0.9193','17,557,581'])
table.add_row(['12','Xception','0.9161','20,817,197'])
table.add_row(['13','Ensemble','0.9314','-'])

print(table)
```

```
+-------+-------------+-----------+----------------------+
| Sr.No |    Model    | val_kappa | Trainable Parameters |
+-------+-------------+-----------+----------------------+
|   1   |   Baseline  |   0.6931  |       1,86,037       |
|   2   |   AlexNet   |   0.7758  |     512,332,165      |
|   3   |    VGG16    |   0.8931  |      9,441,797       |
|   4   |    VGG19    |   0.9128  |      16,521,221      |
|   5   |   DenseNet  |   0.923   |      6,958,981       |
|   6   |   Resnet50  |   0.9091  |      23,544,837      |
|   7   |  Resnet152  |   0.9146  |      58,229,765      |
|   8   | Inceptionv3 |   0.9014  |      21,778,597      |
|   9   | EfficientB0 |   0.9144  |      4,013,953       |
|   10  | EfficientB3 |   0.9199  |      10,703,917      |
|   11  | EfficientB4 |   0.9193  |      17,557,581      |
|   12  |   Xception  |   0.9161  |      20,817,197      |
|   13  |   Ensemble  |   0.9314  |          -           |
+-------+-------------+-----------+----------------------+
```

In [ ]:

/