T.C.
FIRAT ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ
YAZILIM MÜHENDİSLİĞİ BÖLÜMÜ

Projenin Raporlanması

Manisa Şehiri Temiz Hava Tahmini

9.Hafta - Rapor

Proje Ekibi

Erdinç DAĞLI

Haziran 2020

Bu hafta yapmış olduğum

Araştırmalarım:

- Modelleme sonrası oluşturulması hedeflenen doğrulama kısmını için araştırmalar yapılmıştır.
- Feature oluşturma ve metrikleri düzenleme gibi konuları dikkate almak adına bu konularda araştırmalar yaptım.

Yaptıklarım:

- Feature oluşturma kısmındaki metriklerin düzenlenmesi gibi kontroller gerçekleştirmiştir.
- Seçmiş olduğum modellemede şu gibi kavramalardan yararlanmak istedim:
  - Conv1D
  - MaxPooling1D
  - Dropout
  - Flatten
  - Dense
- Standart epok değerini bulmak için test denemeleri yaptım.
- Modelin oluşturduğumuzda
  - Optimezer = “adam”
  - Loss =”binary_crossentropy”
- Bunlara ek olarak ise F1,Kesinlik, Hassasiyet gibi kavramlarında hesaplanmasını sağlayacak metriklerinde girilmesini sağladık.
- Ortalama sonuçları ise görmek adına tek tek sonuçları yazdırdık.
- En son işlem olarak ise kodlamalarımı github a commitledim.

Kod bloklarım:

```
#Model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense,Activation,SimpleRNN,Dropout,MaxPooling1D,Flatten,BatchNormalization,Conv1D
import tensorflow as tf

model = Sequential()
model.add(Conv1D(512,1,input_shape=(nb_features,1)))
model.add(Activation("relu"))
model.add(MaxPooling1D(2))
model.add(Conv1D(256,1))
model.add(Activation("relu"))
model.add(MaxPooling1D(2))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(2048,activation="relu"))
model.add(Dense(1024,activation="relu"))
model.add(Dense(nb_classes,activation="sigmoid"))
model.summary()


from keras import backend as K

def recall_m(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
    recall = true_positives / (possible_positives + K.epsilon())
    return recall

def precision_m(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
    precision = true_positives / (predicted_positives + K.epsilon())
    return precision

def f1_m(y_true, y_pred):
    precision = precision_m(y_true, y_pred)
    recall = recall_m(y_true, y_pred)
    return 2*((precision*recall)/(precision+recall+K.epsilon()))
```

Şekil 1 : Modelin ve diğer fonksiyonların tanımlanması