

## Read Me

```
[ ] # Install required libraries
!pip install -U kaleido
!pip install adjustText
!pip install tensorflow
!pip install hyperopt

from IPython import display
display.clear_output()
```

Pertama tama harus menginstal libraries terlebih dahulu Ditunggu sehingga Semua terinstal Baru lanjut ke step selanjutnya.

### UNSW-NB15: DATA UNDERSTANDING

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

Mounted at /content/drive

Lalu run Sorcecode ini untuk memanggil data set yang kita perlukan seperti

UNSW\_NB15\_testing-set.csv

UNSW\_NB15\_training-set.csv

```
# Baca dataset
df_train = pd.read_csv('/content/drive/MyDrive/IoT-NIDS-Project/Dataset/UNSW_NB15_training-set.csv')
# Lihat 5 sampel data
df_train.head()
```

```
# Baca dataset
df_test = pd.read_csv('/content/drive/MyDrive/IoT-NIDS-Project/Dataset/UNSW_NB15_testing-set.csv')
# Lihat 5 sampel data
df_test.head()
```

Run satu persatu sorcode ini untuk membaca dan memberi sample data traning dan data testing

Setelah itu masuk ke code Hyper parameter di sini kita melakukan satu persatu misalnya jika mengerjakan hyperparameter LSTM Terlebih dahulu.

```
# Gabungkan data train dan test menjadi 1 dataframe
df_combined = pd.concat([df_train, df_test], ignore_index=True)
df_combined
```

Run untuk menggabungkan data training dan data testing menjadi dataframe

Dan untuk sorcode berikutnya di run satu persatu dan jangan run code sekaligus karena bisa menyebabkan error pada laptop karena membutuhkan waktu yang lama untuk satu persatu di run

```
Hyperparameter tuning
Hyperopt
Hyperparameter LSTM
Hyperparameter RNN
Hyperparameter CNN
Evaluation
Training and Evaluation LSTM Using Best Parameters
Training and Evaluation RNN Using Best Parameters
Training and Evaluation CNN Using Best Parameters
Best hyperparameters:
Cross validation
Section
```

```
def tuningLSTM(params):
    # Initial model
    model = Sequential()
    # Jumlah layer
    num_layers = int(params['num_layers'])
    # Tambahkan layer input
    model.add(LSTM(units=params['units'], input_shape=(X_train.shape[1], 1), return_sequences=True))
    # Tambahkan layer dengan dropout
    model.add(Dropout(params['dropout']))
    # Tambahkan layer
    for i in range(num_layers - 1):
        model.add(LSTM(units=int(params['units']), return_sequences=True))
        # Tambahkan layer dengan dropout
        model.add(Dropout(params['dropout']))
    # Tambahkan layer output
    model.add(LSTM(units=int(params['units']), return_sequences=False))
    model.add(Dense(units=2, activation='softmax')) # Menggunakan Dense biasa
    # Compile model
    model.compile(loss='categorical_crossentropy', optimizer=keras.optimizers.RMSprop(learning_rate=params['lr']), metrics=['accuracy'])
    # Summary model
    model.summary()
    # Training model
    history = model.fit(X_train_LSTM, y_train, epochs=params['epochs'], batch_size=1024,
                        validation_data=(X_val_LSTM, y_val), callbacks=[keras.callbacks.EarlyStopping(monitor='val_loss', patience=5)])
    acc = -history.history['val_accuracy'][-1]
    return (float)acc, float(params['STATES']['model'], model)
```

```
[ ] # Fungsi untuk menjalankan training menggunakan model LSTM dengan parameter terbaik
def IoT_NIDS_LSTM():
    model = Sequential()
    # Jumlah layer
    num_layers = 4
    # Tambahkan layer input
    model.add(LSTM(32, input_shape=(X_train.shape[1], 1), return_sequences=True))
    # Tambahkan layer dengan dropout
    model.add(Dropout(0.2))
    # Tambahkan layer
    for i in range(num_layers - 1):
        model.add(LSTM(32, return_sequences=True))
        # Tambahkan layer dengan dropout
        model.add(Dropout(0.2))
    # Tambahkan layer output
    model.add(LSTM(units=32, return_sequences=False))
    model.add(Dense(units=2, activation='softmax')) # Menggunakan Dense biasa
    # Compile model
    model.compile(loss='categorical_crossentropy', optimizer=keras.optimizers.RMSprop(learning_rate=0.0001), metrics=['accuracy'])
    # Summary the model
    model.summary()
    return model

# Create the model
model_LSTM = IoT_NIDS_LSTM()

start = time.time()
# Train the model
history_model_LSTM = model_LSTM.fit(X_train_LSTM, y_train, epochs=100, batch_size=1024, validation_data=(X_val_LSTM, y_val), verbose=1,
```

Kita tunggu hasil dari Hyperparameter LSTM Kkeluar baru kita lanjut run code **Training and Evaluation LSTM Using Best Parameters**. Lalu akan muncul hasil dari validasi Deep learnig LSTM.

Dan begitu juga untuk CNN DAN RNN secara bergantian melakukan run codenya.