Laporan Tugas Implementasi Algorima CNN

Dosen Pengampu:

TJOKORDA AGUNG BUDI WIRAYUDA, S.T., M.T.

Disusun oleh:

Farhan Tirta Kesumah – 1301204108

Telkom University
Fakultas Infmatika
S1 Informatika
2022

Kata Pengantar

Puji dan syukur kehadirat Allah SWT yang telah memberikan rahmat kepada kita untuk menyelesaikan tugas dengan benar dan tepat waktu untuk mata kuliah Pengantar kecerdasan buatan dengan topik "teknik pembelajaran mesin supervised learning".

Tujuan kami mengerjakan Case-Based 1 dan penyusunan laporan ini adalah untuk memenuhi tugas pada mata kuliah Pembelajaran Mesin. Selain itu, laporan ini juga bertujuan untuk menambah wawasan tentang supervised learning bagi para pembaca dan juga bagi penulis.

Kami mengucapkan terima kasih kepada Bapak TJOKORDA AGUNG BUDI WIRAYUDA, S.T., M.T., selaku dosen Pembelajaran Mesin yang telah memberikan tugas ini sehingga dapat menambah ilmu pengetahuan dan wawasan sesuai pada bidang studi kami saat ini.

Kami menyadari, laporan yang kami tulis ini masih jauh dari kata sempurna. Maka dari itu kami beinisiatif untuk berusaha berkembang dan menjadi lebih baik kedepannya.

BAB I
PENDAHULUAN

1. Persoalan

1. 1. Definisi Tugas

Anda diminta untuk melakukan beberapa analisis dan menghasilkan seperangkat aturan yang berguna menggunakan dataset berikut: *https://archive.ics.uci.edu/ml/datasets/Arrhythmia.*

Selidiki masalah kualitas data yang telah diberikan di atas. Jelaskan keputusan Anda mengenai pendekatan pra-pemrosesan data. Jelajahi kumpulan data dengan meringkas data menggunakan statistik dan mengidentifikasi masalah kualitas data apa pun. Tidak ada batasan jumlah ringkasan yang akan dilaporkan tetapi Anda diharapkan hanya melaporkan yang paling relevan.

Pilih salah satu dari metode unsupervised learning yang telah dipelajari yaitu AAN/MLP/RNN/LSTM/CNN. Anda hanya perlu memilih satu metode untuk diterapkan. Gunakan algoritma tersebut untuk memberikan beberapa output/outcome dengan menggunakan variasi hyperparameter, kemudian menganalisis hasilnya.

1. 2. Output Program

Menuliskan evaluasi terkait hasil yang Anda (atau belum) temukan, buat dalam bentuk tabel. Kemudian, jelaskan pengukuran performansi yang Anda lakukan dan berikan analisis mendalam mengenai hasil yang diperoleh.

BAB II
PEMBAHASAN

2. 1. *Load Data*

```
[2]  # Load data dari repository github
     dataset = pd.read_excel('https://raw.githubusercontent.com/farhantk/ML-case-based-1/main/arrhythmia.xlsx')
     dataset.shape

     (452, 280)
```

Data disimpan pada *repository* github dengan menggunakan *library* pandas, diatas terlihat bahwa terdapat 452 baris dan 280 kolom.

2. 2. *Preprocessing*

```
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import StandardScaler
#Preprocessing
# pisahkan data X dan Y(target)
X = dataset.iloc[:,:-1]
Y = dataset.iloc[:,-1]

# Hapus kolom yang memiliki data kosong sebanyak 30%
thresh = len(X) * 0.3
X.dropna(thresh = thresh, axis = 1, inplace = True)

# Jika ada data yang bernilai null akan diisi dengan mediannya
mean = SimpleImputer(missing_values=np.NaN, strategy='median')
imputer = mean.fit(X)
df_imp = imputer.transform(X)
X = pd.DataFrame(df_imp)

# Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_scaled = sc.fit_transform(X.values)
X = pd.DataFrame(x_scaled, index = X.index)

X.shape
```

Pada tahap ini dari data yang sudah ada dipisahkan terlebih dahulu dengan targetnya, lalu jika ada *value* pada kolom yang kosong sebanyak 30%, kolom tersebut akan dihapus. Jika ada data yg benilai null akan diganti dengan medianya. Scalling dilakukan agar data berada pada rentang 0 hingga 1. Dari proses ini kolom berkurang menjadi 278.

2. 3. *Split Data*

```
# Memisahkan data train dan data test
from keras.utils import to_categorical
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3,  random_state=5)

Xtrain = np.array(X_train)
Ytrain = to_categorical( Y_train)
Xtest = np.array(X_test)
Ytest = to_categorical(Y_test)
```

Proses ini dilakukan dengan membagi dua data sebesar 70% untuk *train* dan 30% untuk *test* untuk mengetahui akurasinya dengan mengvalidasi hasil data t*rain* pada data *test.*

3. 4. *Ketidakseimbangan Data*

```
from sklearn.utils import class_weight
mlp_class_weights = class_weight.compute_class_weight(class_weight = 'balanced', classes = np.unique(Y_train), y = Y_train)
mlp_class_weights_dict = dict(zip([1,2,3,4,5,6,7,8,9,10,14,15,16], mlp_class_weights))
mlp_class_weights_dict[0] = 0
mlp_class_weights_dict[11] = 0
mlp_class_weights_dict[12] = 0
mlp_class_weights_dict[13] = 0
print(mlp_class_weights.sum())
print(mlp_class_weights_dict)
```

Bila kita melihat target(Y) terdapat nilai yang lebih dominan atau mayoritas seperti 1, 15, dan 16 jika dibandingkan dengan target lainya. Pada proses ini bobot data mayoritas akan dikurangi otomatis bobot data minoritas akan lebih tinggi.

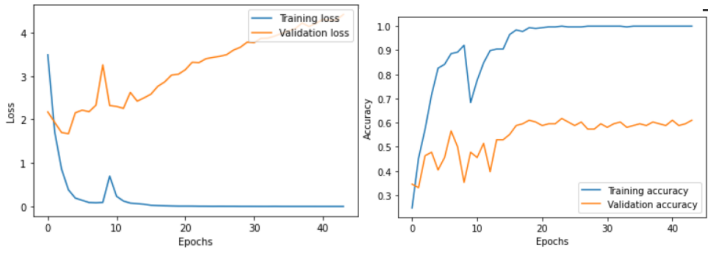
2. 5. Model CNN

Pada prosess ini dibuat beberapa model CNN dengan fungsi aktivasi dan *optimizer* yang berbeda-beda, seperti dibawah:

1.

```
from keras.layers.convolutional import Conv1D
from keras.layers import Activation, Dense , Dropout, Flatten
from keras.models import Sequential
#model
CNN = Sequential()

CNN.add(Conv1D(filters=64, kernel_size=10,activation='relu',kernel_initializer='he_uniform', input_shape=(278,1)))
CNN.add(Conv1D(filters=128, kernel_size=10,activation='relu',kernel_initializer='he_uniform'))
CNN.add(Dropout(0.3))
CNN.add(Dense(64, activation='relu'))
CNN.add(Dense(32, activation='relu'))
CNN.add(Flatten())
CNN.add(Dense(17, activation='softmax'))
CNN.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

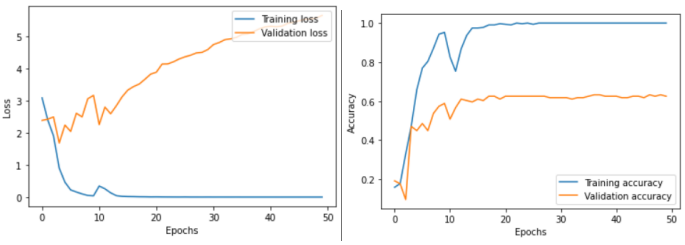


Akurasi training : 100 %
Akurasi Validasi : 61 %

2.

```
from keras.layers.convolutional import Conv1D
from keras.layers import Activation, Dense , Dropout, Flatten
from keras.models import Sequential
#model
CNN = Sequential()

CNN.add(Conv1D(filters=64, kernel_size=10,activation='relu',kernel_initializer='he_uniform', input_shape=(278,1)))
CNN.add(Conv1D(filters=128, kernel_size=10,activation='relu',kernel_initializer='he_uniform'))
CNN.add(Dropout(0.3))
CNN.add(Dense(128, activation='relu'))
CNN.add(Dense(64, activation='relu'))
CNN.add(Dense(32, activation='relu'))
CNN.add(Flatten())
CNN.add(Dense(17, activation='softmax'))
CNN.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

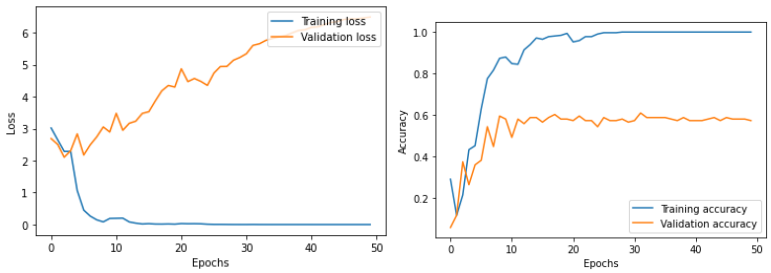


Akurasi training : 100 %
Akurasi Validasi : 62,5 %

3.

```
from keras.layers.convolutional import Conv1D
from keras.layers import Activation, Dense , Dropout, Flatten
from keras.models import Sequential
#model
CNN = Sequential()

CNN.add(Conv1D(filters=64, kernel_size=10,activation='relu',kernel_initializer='he_uniform', input_shape=(278,1)))
CNN.add(Conv1D(filters=128, kernel_size=10,activation='relu',kernel_initializer='he_uniform'))
CNN.add(Dropout(0.3))
CNN.add(Dense(256, activation='relu'))
CNN.add(Dense(128, activation='relu'))
CNN.add(Dense(64, activation='relu'))
CNN.add(Dense(32, activation='relu'))
CNN.add(Flatten())
CNN.add(Dense(17, activation='softmax'))
CNN.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```



Akurasi training : 100 %
Akurasi Validasi : 57,35 %

2. 6. Evaluasi

Dari metode diatas dengan epoch 50 dapat dilihat bahwa akurasi mulai tidak meningkat saat epoch ke-22 dan loss validation training mulai tidak ada perubahan pada epoch ke-22. Hal ini sejalan dengan akurasi yang mulai tidak berubah juga pada epoch ke-22. Dari ketiganyapun tidak ditemukan perbedaan yang signifikan.

Dari grafik yang ada terlihat kesamaan bahwa jarak antara akurasi training dan validasi yang jauh, hal ini bisa dinilai sebagai *overfiting.*

BAB III
PENUTUP

3. 1. Kesimpulan

Dari percobaan metode CNN dengan data arrhythmia didapat akurasi 100% pada training dan 62,5% validasi. Dengan perbedaan yang sangat signifikan antara akurasi training dan validasi, maka bisa disimpulkan terjadi *overfitting*.

3. 2. Lampiran

Google Colab:
https://colab.research.google.com/drive/1vg60xL8ePbR810CfHUjfdVNQL6YJAmCw?usp=sharing
Video: *https://youtu.be/owdu9kk7XSU*

DAFTAR PUSTAKA

https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/

https://www.analyticsvidhya.com/blog/2020/10/improve-class-imbalance-class-weights/

https://www.linkedin.com/pulse/100-accuracy-supremacy-imperfection-overfitting-vs-utkarsh-sharma