Dr. Ir. Kurnianingsih, S.T., M.T. – Prodi Teknologi Rekayasa Komputer, Politeknik Negeri Semarang

Jobsheet 9

NEURAL NETWORK

A. Tujuan

1. Praktikan mampu memahami konsep dasar jaringan syaraf tiruan
2. Praktikan mampu memahami langkah-langkah jaringan syaraf tiruan

B. Landasan Teori

Jaringan saraf tiruan (JST) (Bahasa Inggris: artificial neural network (ANN), atau juga disebut simulated neural network (SNN), atau umumnya hanya disebut neural network (NN)), adalah jaringan dari sekelompok unit pemroses kecil yang dimodelkan berdasarkan jaringan saraf manusia. JST merupakan sistem adaptif yang dapat mengubah strukturnya untuk memecahkan masalah berdasarkan informasi eksternal maupun internal yang mengalir melalui jaringan tersebut.

Secara sederhana, JST adalah sebuah alat pemodelan data statistik non-linier. JST dapat digunakan untuk memodelkan hubungan yang kompleks antara input dan output untuk menemukan pola-pola pada data.

Perbandingan Jaringan Syaraf Tiruan dengan Konvensional

Jaringan Syaraf Tiruan memiliki pendekatan yang berbeda untuk memecahkan masalah bila dibandingkan dengan sebuah komputer konvensional. Umumnya komputer konvensional menggunakan pendekatan algoritma (komputer konvensional menjalankan sekumpulan perintah untuk memecahkan masalah). Jika suatu perintah tidak diketahui oleh komputer konvensional maka komputer konvensional tidak dapat memecahkan masalah yang ada.

Sangat penting mengetahui bagaimana memecahkan suatu masalah pada komputer konvensional dimana komputer konvensional akan sangat bermanfaat jika dapat melakukan sesuatu dimana pengguna belum mengatahui bagaimana melakukannya. Jaringan Syaraf Tiruan dan suatu algoritma komputer konvensional tidak saling bersaing namun saling melengkapi satu sama lain. Pada suatu kegiatan yang besar, sistim yang diperlukan biasanya menggunakan kombinasi antara keduanya (biasanya sebuah komputer konvensional digunakan untuk mengontrol Jaringan Syaraf Tiruan untuk menghasilkan efisiensi y ang maksimal. Jaringan Syaraf Tiruan tidak memberikan suatu keajiban tetapi jika digunakan secara tepat akan menghasilkan sasuatu hasil yang luar biasa.

Cara Kerja Jarinngan Saraf Tiruan

1. Supervised Learning (pembelajaran terawasi)
2. Unsupervised Learning (pembelajawan tidak terawasi)
3. Reinforced Learning

Aplikasi Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan mampu menggambarkan setiap situasi adanya sebuah hubungan antara variabel predictor (independents, inputs) dan variabel predicted (dependents, outputs), ketika hubungan tersebut sangat kompleks dan tidak mudah untuk menjelaskan kedalam istilah yang umum dari “correlations” atau “differences between groups”. Beberapa contoh permasalahan yang dapat dipecahkan secara baik oleh Jaringan Syaraf Tiruan antara lain :

1. Deteksi Fenomena kedokteran. Berbagai indikasi yang berhubungan dengan kesehatan (kombinasi dari denyut jantung, tingkatan dan berbagai substansi dalam darah, dll) dapat dimonitoring. Serangan pada kondisi kesehatan tertentu dapat dihubungan dengan perubahan kombinasi yang sangat kompeks (nonlinear dan interaktif) pada subset dari variabel, dapat dimonitoring. Jaringan Syaraf Tiruan telah digunakan untuk mengenali pola yang diperkirakan sehingga perlakuan yang tepat dapat dilakukan

2. Untuk mendeteksi golongan darah manusia Dengan menggunakan pengolahan citra. Manusia berusaha keras dengan segala kemampuannya untuk menirukan kehebatan yang mereka miliki, misalnya seorang dokter dengan keahliannya dapat membedakan golongan darah manusia antara A, B, AB, dan O. Dengan pendekatan kecerdasan buatan, manusia berusaha menirukan bagaimana pola- pola dibentuk. Jaringan Syaraf Tiruan telah dikembangkan sebagai generalisasi model matematik dari pembelajaran manusia
3. Prediksi Pasar Saham. Fluktuasi dari harga saham dan index saham adalah contoh lain yang kompleks, multidimesi tetapi dalam beberapa kondisi tertentu merupakan phenomena yang dapat prediksi. Jaringan Syaraf Tiruan telah digunakan oleh analis teknik untuk membuat prediksi tentang pasar saham yang didasarkan atas sejumlah faktor seperti keadaan masa lalu bursa yang lain dan berbagai indikator ekonomi
4. Perjanjian Kredit. Berbagai informasi biasanya didapat dari seorang peminjam seperti umur, pendidikan, pekerjaan dan berbagai data lain. Setelah pembelajaran dari Jaringan Syaraf Tiruan tentang data peminjam, analisis Jaringan Syaraf Tiruan dapat mengidentifikasi karaktersetik peminjam sehingga dapat digunakan untuk mengklasifikasikan peminjam terhadap resiko peminjam dalam kategori baik atau buruk
5. Monitoring Kondisi Mesin. Jaringan Syaraf Tiruan dapat digunakan untuk memangkas biaya dengan memberikan keahlian tambahan untuk menjadwalkan perawatan mesin. Jaringan syaraf tiruan dapat dilatih untuk membedakan suara sebuah mesin ketika berjalan normal (“false alarm”) dengan ketika mesin hampir mengalami suatu masalah. Setelah periode pembelajaran, keahlian dari Jaringan Syaraf Tiruan dapat digunakan untuk memperingatkan seorang teknisi terhadap kerusakan yang akan timbul sebelum terjadi yang akan menyebabkan biaya yang tidak terduga
6. Pemeliharaan Mesin. Jaringan Syaraf Tiruan telah digunakan untuk menganalisis input dari sebuah sensor pada sebuah mesin. Dengan mengontrol beberapa parameter ketika mesin sedang berjalan, dapat melakukan fungsi tertentu misalnya meminimalkan penggunaan bahan bakar

KESIMPULAN

Kesimpulan Jaringan Syaraf Tiruan mulai dilirik banyak kalangan karena mempunyai banyak kelebihan dibandingkan system konvensional. Jaringan Syaraf Tiruan mewakili pikiran manusia untuk mendekatkan diri dengan komputer, maksudnya Jaringan Syaraf Tiruan dirancang agar komputer dapat bekerja seperti/layaknya otak manusia. Berikut ini beberapa keunggulan dari Jaringan Syaraf Tiruan adalah :

1. Adaptive learning: Suatu kemampuan untuk melakukan suatu kegiatan yang didasarkan atas data yang diberikan pada saat pembelajaran atau dari pengalaman sebelumnya.
2. Self-Organisation: Dapat membuat organisasi sendiri atau me-representasikan informasi yang didapat pada saat pembelajaran.
3. Real Time Operation: Dapat menghasilkan perhitungan parallel dan dengan device hardware yang khusus yang dibuat akan memberikan keuntungan dengan adanya kemampuan tersebut.
4. Fault Tolerance melalui Redundant Information Coding: Kerusakan pada bagian tertentu dari jaringan akan mengakibatkan penurunan kemampuan. Beberapa jaringan mempunyai kemampuan untuk menahan kerusakan besar pada jaringan.
5. Kelebihan Jaringan Syaraf Tiruan terletak pada kemampuan belajar yang dimilikinya. Dengan kemampuan tersebut pengguna tidak perlu merumuskan kaidah atau fungsinya. Jaringan Syaraf Tiruan akan belajar mencari sendiri kaidah atau fungsi tersebut. Dengan demikian Jaringan Syaraf Tiruan mampu digunakan untuk menyelesaikan masalah yang rumit dan atau masalah yang terdapat kaidah atau fungsi yang tidak diketahui.
6. Kemampuan Jaringan Syaraf Tiruan dalam menyelesaikan masalah yang rumit telah dibuktikan dalam berbagai macam penelitian.

CONTOH PROGRAM DAN MATLAB

1. Koding untuk menuliskan data latih dan target latih pada matlab

```
% Mempersiapkan data latih dan target latih
data_latih=[1 1;1 0;0 1;0 0]';
kelas_latih=[0 1 1 0];
```

Hasil

```
data_latih =
   1   1   0   0
   1   0   1   0
kelas_latih =
   0   1   1   0
```

2. Membuat plot data latih

```
%Plot data latih

figure('Position',[300 300 210 160]);
plotpv(data_latih,kelas_latih);
point = findobj(gca,'type','line');
set(point,'Color','black');
HL = 2;
```

3. Selanjutnya membuat coding Jaringan Syaraf Tiruan Backpropagation dengan arsitektur 4-2-1 dan inisialisasi bobot awal secara acak. Pada pemrograman ini digunakan fungsi aktivasi sigmoid biner (logsig) pada hidden layer dan fungsi aktivasi linear (purelin) pada layer keluaran. Sedangkan fungsi pelatihan menggunakan metode gradien descent

```
% Pembuatan JST

net = newff(minmax(data_latih),[HL 1], {'logsig' 'logsig'},'traingdx');

net.IW{1,1}=[0.3 0.5; -0.8 0.1];

net.LW {2,1} = [0.2 0.7];

net.b {1,1} = [0.2; -0.2];

net.b{2,1} = [0.3]
```

4. Membuat coding untuk memberikan parameter-parameter yang mempengaruhi proses pelatihan jst seperti parameter jumlah epoch, target error, learning rate, momentum.

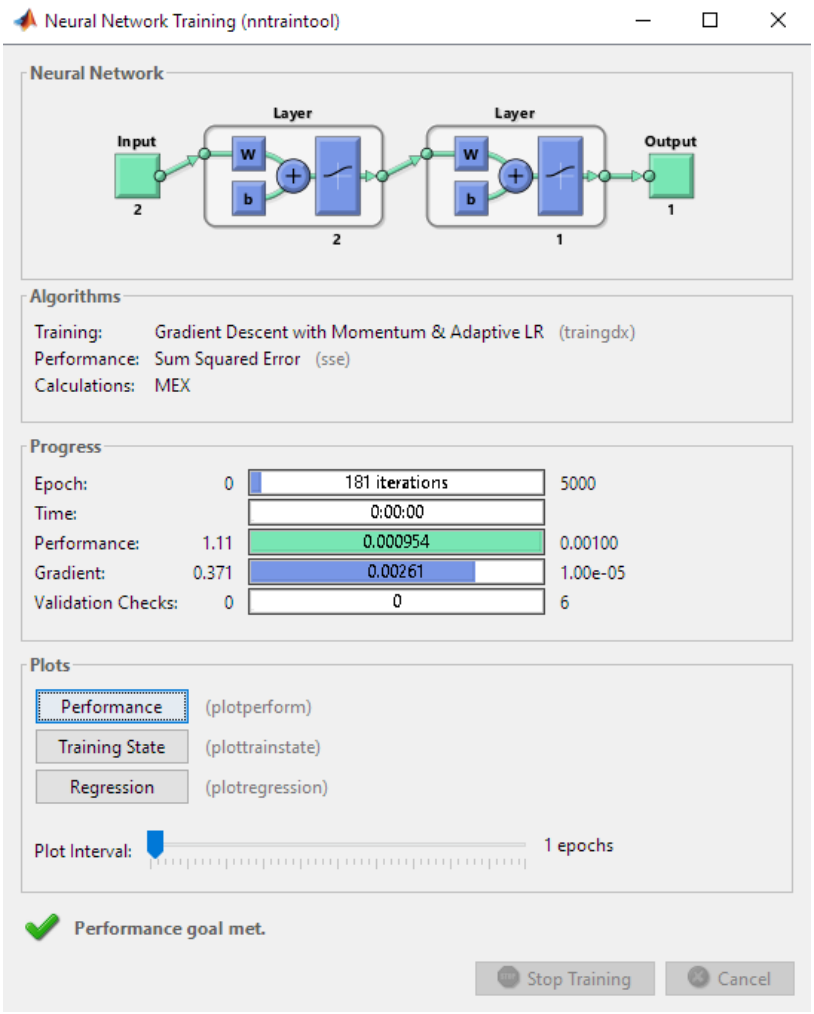
```
% Memberikan nilai untuk mempengaruhi proses pelatihan

net.performFcn = 'sse';

net.trainParam.goal = 0.001;

net.trainParam.show = 20;

net.trainParam.epochs = 5000;

net.trainParam.mc = 0.95;

net.trainParam.lr = 0.1;
```

5. Membuat coding untuk melakukan pelatihan jaringan

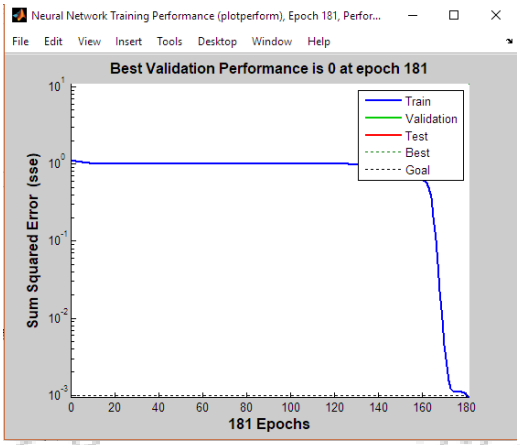
```
% Proses training
```

```
[net_keluaran,tr,Y,E] = train(net,data_latih,target_latih);
```

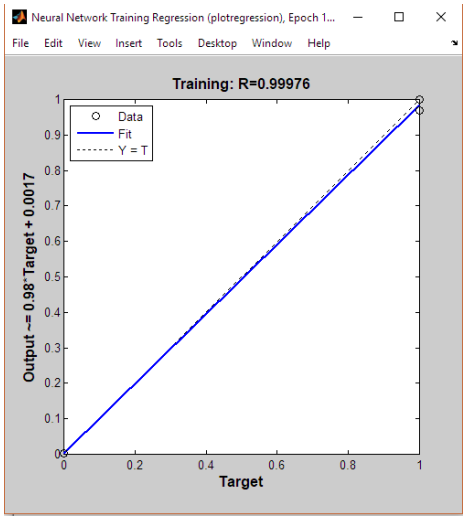
Hasilnya



Pada tampilan tersebut ditunjukkan bahwa target error (sse) tercapai pada epoch ke 181. Kita bisa melihat error (sse) yang dihasilkan pada setiap epoch dengan meng-klik tombol ‘performance’ sehingga muncul tampilan seperti berikut:



Sedangkan koefisien korelasi hasil pelatihan dapat dilihat dengan meng-klik tombol 'regression' sehingga diperoleh:



Nilai koefisien korelasi sebesar 0.99976 menunjukkan bahwa akurasi hasil proses pelatihan baik.

6. Untuk melihat nilai-nilai hasil pelatihan, kita dapat menuliskan coding sbb:

```
%Hasil setelah pelatihan

bobot_hidden=net_keluaran.IW{1,1};
```

```
bobot_keluaran = net_keluaran.LW{2,1}

bias_hidden = net_keluaran.b{1,1}

bias_keluaran = net_keluaran.b{2,1}

jumlah_iterasi=tr.num_epochs

nilai_keluaran = Y

nilai_error=E

error_SSE=0.5*sum(nilai_error.^2)
```

Latihan SLP dan MLP dengan Python

```
import numpy as np
import pandas as pd
from pandas import Series, DataFrame
import seaborn as sns
import matplotlib.pyplot as plt

iris = pd.read_csv("Iris.csv")
iris.head()

iris.info()
iris.drop("Id", axis=1, inplace = True)

fig = iris[iris.Species == 'Iris-
setosa'].plot(kind='scatter', x='SepalLengthCm', y='SepalWidthCm', color='orange', label
='Setosa')
iris[iris.Species == 'Iris-
versicolor'].plot(kind='scatter', x='SepalLengthCm', y='SepalWidthCm', color='blue', lab
el='Versicolor', ax=fig)
iris[iris.Species == 'Iris-
virginica'].plot(kind='scatter', x='SepalLengthCm', y='SepalWidthCm', color='green', lab
el='Virginica', ax=fig)

fig.set_xlabel('Sepal Length')
fig.set_ylabel('Sepal Width')
fig.set_title('Sepal Length Vs Width')

fig=plt.gcf()
```

```
fig.set_size_inches(10, 7)
plt.show()
sns.FacetGrid(iris, hue='Species', size=5)   .map(plt.scatter, 'SepalLengthCm', 'SepalWi
dthCm')   .add_legend()

fig = iris[iris.Species == 'Iris-
setosa'].plot(kind='scatter', x='PetalLengthCm', y='PetalWidthCm', color='orange', label
='Setosa')
iris[iris.Species == 'Iris-
versicolor'].plot(kind='scatter', x='PetalLengthCm', y='PetalWidthCm', color='blue', lab
el='Versicolor', ax=fig)
iris[iris.Species == 'Iris-
virginica'].plot(kind='scatter', x='PetalLengthCm', y='PetalWidthCm', color='green', lab
el='Virginica', ax=fig)

fig.set_xlabel('Petal Length')
fig.set_ylabel('Petal Width')
fig.set_title('Petal Length Vs Width')

fig=plt.gcf()
fig.set_size_inches(10, 7)
plt.show()

iris.hist(edgecolor='black', linewidth=1.2)
fig = plt.gcf()
fig.set_size_inches(12,6)
plt.show()

plt.figure(figsize=(15,10))
plt.subplot(2,2,1)
sns.violinplot(x='Species', y = 'SepalLengthCm', data=iris)
plt.subplot(2,2,2)
sns.violinplot(x='Species', y = 'SepalWidthCm', data=iris)

plt.subplot(2,2,3)
sns.violinplot(x='Species', y = 'PetalLengthCm', data=iris)
plt.subplot(2,2,4)
sns.violinplot(x='Species', y = 'PetalWidthCm', data=iris)

# importing alll the necessary packages to use the various classification algorithms
from sklearn.linear_model import LogisticRegression # for Logistic Regression Algorithm
from sklearn.model_selection import train_test_split # to split the dataset for training
 and testing
from sklearn.neighbors import KNeighborsClassifier # KNN classifier
from sklearn.neural_network import MLPClassifier # for neural network
from sklearn import metrics # for checking the model accuracy
from sklearn.tree import DecisionTreeClassifier # for using DTA
```

```
iris.shape
plt.figure(figsize=(8,4))
sns.heatmap(iris.corr(), annot=True, cmap='cubehelix_r') # draws heatmap with input as c
orrelation matrix calculated by iris.corr()
plt.show()

# ## Splitting The Data into Training And Testing Dataset
train, test = train_test_split(iris, test_size=0.3) # our main data split into train and
 test
# the attribute test_size=0.3 splits the data into 70% and 30% ratio. train=70% and test
=30%
print(train.shape)
print(test.shape)

train_X = train[['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm']]
train_y = train.Species # output of the training data

test_X = test[['SepalLengthCm','SepalWidthCm','PetalLengthCm','PetalWidthCm']] # taking
test data feature
test_y = test.Species # output value of the test data

train_X.head()
test_X.head()

train_y.head()

petal = iris[['PetalLengthCm','PetalWidthCm','Species']]
sepal = iris[['SepalLengthCm','SepalWidthCm','Species']]

train_p,test_p = train_test_split(petal, test_size=0.3, random_state=0) #petals
train_x_p = train_p[['PetalWidthCm','PetalLengthCm']]
train_y_p = train_p.Species

test_x_p = test_p[['PetalWidthCm','PetalLengthCm']]
test_y_p = test_p.Species
train_s,test_s = train_test_split(sepal, test_size=0.3, random_state=0) #sepals
train_x_s = train_s[['SepalWidthCm','SepalLengthCm']]
train_y_s = train_s.Species

test_x_s = test_s[['SepalWidthCm','SepalLengthCm']]
test_y_s = test_s.Species

# ## MLP Algorithm
model=MLPClassifier(hidden_layer_sizes=(15,15,15), random_state=1, max_iter=150, warm_st
art=True)
model.fit(train_x_p,train_y_p)
prediction=model.predict(test_x_p)
print('The accuracy of the MLP using Petals is:',metrics.accuracy_score(prediction,test_
y_p))
```

C. Instruksi Praktikum

1. Buka file JST.m running menggunakan matlab capture hasilnya.
2. Ubahlah jumlah hidden layer yang digunakan dengan 3, 5 dan 7 layer amati perbedaannya dan berikan analisa
3. Running file MultiLayerPerceptron pada python. untuk memahami pemrograman jaringan syaraf tiruan pada bahasa pemrograman python
4. Buatlah MLP dengan berbagai fungsi aktivasi untuk dataset Ionosphere.csv