Praktikum 5
Membuat Program Aplikasi Backpropagation

A. Tujuan

1. Mahasiswa dapat memahami dan menjelaskan konsep Backpropagation
2. Mahasiswa dapat menjelaskan model Backpropagation
3. Mahasiswa dapat membuat aplikasi Backpropagation

Software yang diperlukan:

- Microsoft Visual C++
- PyCharm

B. Pendahuluan

1. Backpropagation

Jaringan Syaraf Tiruan (JST) merupakan salah satu sistem pemrosesan informasi atau data yang didisain dengan menirukan cara kerja otak manusia dalam menyelesaikan suatu masalah dengan melakukan proses belajar melalui perubahan bobot sinapsisnya. JST yang berupa susunan sel-sel saraf tiruan (neuron) dibangun berdasarkan prinsip-prinsip organisasi otak manusia. Salah satu metode yang digunakan dalam JST adalah Backpropagation.

Backpropagation adalah algoritma pembelajaran untuk memperkecil tingkat error dengan cara menyesuaikan bobotnya berdasarkan perbedaan output dan target yang diinginkan. Backpropagation juga merupakan sebuah metode sistematik untuk pelatihan multilayer JST. Backpopagation dikatakan sebagai algoritma pelatihan multilayer karena Backpropagation memiliki tiga layer dalam proses pelatihannya, yaitu input layer, hidden layer dan output layer, dimana backpropagation ini merupakan perkembangan dari *single layer network* (Jaringan Layar Tunggal) yang memiliki dua layer, yaitu input layer dan output layer. Dengan adanya hidden layer pada backpropagation dapat menyebabkan besarnya tingkat error pada backpropagation lebih kecil dibanding tingkat error pada single layer network. Hal tersebut dikarenakan hidden layer pada backpropagation berfungsi sebagai tempat untuk mengupdate dan menyesuaikan bobot, sehingga

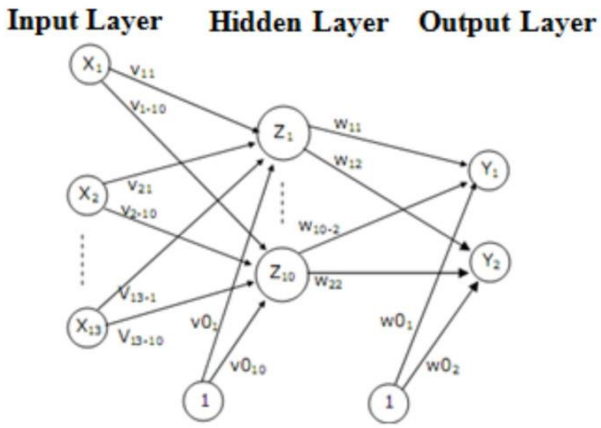
didapatkan nilai bobot yang baru yang bisa diarahkan mendekati dengan target output yang diinginkan.

Merupakan sebuah metode yang dapat melakukan pembelajaran maju dan mundur berulang-ulang untuk mendapatkan bobot yang terbaik. Perhitungan maju digunakan untuk melakukan perhitungan error antara output aktual dan target. Perhitungan mundur digunakan untuk memperbaiki bobot – bobot sinaptik pada semua neuron yang ada.

Backpropagation (BP) merupakan JST multi-layer. Penemuannya mengatasi kelemahan JST dengan layer tunggal yang mengakibatkan perkembangan JST sempat tersendat disekitar tahun 1970. Algoritma BP merupakan generalisasi aturan delta (Widrow-Hoff), yaitu menerapkan metode gradient descent untuk meminimalkan error kuadrat total dari keluaran yang dihitung oleh jaringan. Banyak aplikasi yang dapat diselesaikan dengan BP, akibatnya JST semakin banyak diminati orang. JST layer tunggal memiliki kelemahan dalam pengenal pola. Hal ini di atasi dengan menambah satu/beberapa layer tersembunyi diantara layar masukan dan layer keluaran. Banyak layar tersembunyi memiliki kelebihan manfaat untuk beberapa kasus, namun pelatihannya memerlukan waktu yang lama. Sesuai dengan ide dasar JST, BP melatih jaringan untuk memperoleh keseimbangan antara “kemampuan jaringan” untuk mengenali pola yang digunakan selama pelatihan dan “kemampuan jaringan” merespon secara benar terhadap pola masukan yang serupa (tapi tidak sama) dengan pola pelatihan.

2. Arsitektur Backpropagation

Arsitektur algoritma backpropagation terdiri dari tiga layer, yaitu input layer, hidden layer dan output layer. Pada input layer tidak terjadi proses komputasi, namun pada input layer terjadi pengiriman sinyal input X ke hidden layer. Pada hidden dan output layer terjadi proses komputasi terhadap bobot dan bias dan dihitung pula besarnya output dari hidden dan output layer tersebut berdasarkan fungsi aktivasi tertentu. Dalam algoritma backpropagation ini digunakan fungsi aktivasi sigmoid biner, karena output yang diharapkan bernilai antara 0 sampai 1.



Gambar 1. Arsitektur Backpropagasi

- Tiga layer backpropagation adalah input layer, hidden layer dan output layer.
- Pada input layer, inputan divariabelkan dengan Xn.
- Pada hidden layer, terdapat bobot (Vij) dan bias (Voj), serta Z sebagai data hidden layer.
- Pada output layer juga demikian, terdapat bobot (Wij) dan bias (Woj) dengan data output divariabelkan dengan Y.

Algoritma backpropagation adalah sebuah algoritma untuk memperkecil tingkat error dengan menyesuaikan bobot berdasarkan perbedaan output dan target yang diinginkan. Secara umum algoritmanya terdiri dari tiga langkah utama, yaitu :

- Pengambilan input
- Penelusuran error
- Penyesuaian bobot

Pada pengambilan input, terlebih dahulu dilakukan inisialisasi bobot, kemudian masuk ke dalam algoritma proses backpropagation yang terdiri dari komputasi maju yang bertujuan untuk menelusuri besarnya error dan komputasi balik untuk mengupdate dan menyesuaikan bobot.

Dalam mengupdate bobot dapat dilakukan dengan dua cara, yaitu tanpa momentum dan dengan momentum. Namun, yang dijelaskan di bawah ini dalam mengupdate bobotnya dilakukan tanpa memperhatikan besarnya momentum. Dengan demikian dalam metode backpropagation, algoritma yang harus dilakukan adalah inisialisasi bobot, komputasi feed forward dan backpropagation dan inisialisasi kondisi stopping berdasarkan nilai batas error atau jumlah batas epoch. Epoch merupakan rangkaian langkah dalam pembelajaran ANN. Satu epoch diartikan sebagai satu kali pembelajaran ANN.

3. Aplikasi Backpropagation

1. Pengenalan Kapal pada Citra Digital Menggunakan Image Processing dengan Jaringan Syaraf Tiruan Backpropagation
2. Prediksi Nilai Tukar Petani Menggunakan Jaringan Syaraf Tiruan Backpropagation
3. Klasifikasi Pengendara Sepeda Motor tidak Memakai Helm pada Citra Digital dengan Jaringan Syaraf Tiruan Backpropagation
4. Aplikasi Prediksi Jumlah Penderita Penyakit Demam Berdarah Dengue menggunakan Jaringan Syaraf Tiruan Backpropagation
5. Pengendalian Sudut Arah Mobile Robot Menggunakan Jaringan Syaraf Tiruan Backpropagation
6. Persoalan klasifikasi untuk penerimaan pegawai

4. Contoh Implementasi Aplikasi

- Berikut persoalan klasifikasi untuk Penerimaan Pegawai.
- Proses penerimaan pegawai ditentukan oleh 3 atribut, yaitu : IPK, Psikologi dan Wawancara
- Atribut IPK memiliki 3 kemungkinan nilai : bagus, cukup, kurang
- Atribut Psikologi memiliki 3 kemungkinan nilai : tinggi, sedang, rendah
- Atribut Wawancara memiliki 2 kemungkinan nilai : baik dan buruk.

- Gunakan Data berikut sebagai Data Training

Pelamar	IPK	Psikologi	Wawancara	Diterima
1	3	3	2	1
2	3	2	2	1
3	3	2	1	1
4	3	1	1	0
5	2	3	2	1
6	2	2	2	1
7	2	2	1	1
8	2	1	1	0
9	1	3	2	1
10	1	2	1	0
11	1	1	2	1

- Gunakan data berikut sebagai data testing

Pelamar	IPK	Psikologi	Wawancara	Diterima
12	3	3	1	?
13	3	1	2	?
14	2	3	1	?
15	2	1	2	?
16	1	3	1	?
17	1	2	2	?
18	1	1	1	?

- Misal kita gunakan arsitektur 3-2-1, itu artinya 3 node pada input layer, 2 node pada hidden layer dan 1 node pada output layer.
- Sebagai kondisi berhenti diset nilai MSE = 10^{-5}.
- Maksimum iterasi = 5000 epoch
- Learning rate kita set = 0.1
- Inisialisasi pemberat didapatkan dari bilangan acak, dalam interval [-0.1; +0.1].

  Contoh :

  W1 = [0.0588 -0.08999;
  -0.0882 -0.0169;
  0.0206 -0.0390]

  W2 = [0.0749;
  -0.0970]

- Setelah proses pelatihan selesai, kita bisa melakukan pengujian jaringan dengan menggunakan data testing (pengujian).
- Tujuannya adalah untuk mengetahui apakah proses pelatihan JST sudah benar atau belum.

5. Contoh Program Backpropagation

```
#define _CRT_SECURE_NO_DEPRECATE
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <math.h>
/*---------------------- random function ----------------------*/
float  d_rand( void )
{
  return ((float)(((rand() % 32767) / 32767.0 - 0.5 ) * 2.0)) ;
}
/*-------------------------- sigmoid ------------------------------*/
float  sigmoid(float u)
{
  return ((float)(1.0/(1.0 + exp(-u))));
}
int void()
{
int i,j,p,l;
float z,delta_o,delta_h[6],g1,f1[6];
float y[6]={0.0,0.0,1.0,0.0,0.0,0.0};
float x[11][4]={{1.00,1.00,0.67,1.00},
  {1.00,0.67,0.67,1.00},
  {1.00,0.67,0.33,1.00},
  {1.00,0.33,0.33,1.00},
   {0.67,1.00,0.67,1.00},
  {0.67,0.67,0.67,1.00},
  {0.67,0.67,0.33,1.00},
  {0.67,0.33,0.33,1.00},
  {0.33,1.00,0.67,1.00},
  {0.33,0.67,0.33,1.00},
  {0.33,0.33,0.67,1.00}};
float t[11]={1.0,1.0,1.0,0.0,1.0,1.0,1.0,0.0,1.0,0.0,1.0};
float w[6][4],O[6],s[6],LR=0.1f,init=0.15f,error;
FILE *f;
f=fopen("error.txt","w");
```

```
//inisialisasi bobot
	for(j=0;j<2;j++)
	{
		for(i=0;i<4;i++)
		{
			w[j][i]=init*d_rand();
		}
	}
	for(j=0;j<3;j++)
	{
		s[j]=init*d_rand();
	}

//training
	for(l=0;l<5000;l++)
	{
		error=0.0;
		for(p=0;p<11;p++)
		{
			for(j=0;j<2;j++)
			{
				O[j]=0.0;
				for(i=0;i<4;i++)
				{
					O[j]=O[j]+x[p][i]*w[j][i];
				}
				y[j]=sigmoid(O[j]);
			}
			O[0]=0.0;
			for(i=0;i<3;i++)
			{
				O[0]=O[0]+y[i]*s[i];
			}
			z=sigmoid(O[0]);
```

```
                g1=z*(1-z);
                                delta_o=(t[p]-z)*g1;
                                for(j=0;j<2;j++)
                                {
                                        f1[j]=y[j]*(1-y[j]);
                                }
                                for(j=0;j<2;j++)
                                {
                                        delta_h[j]=f1[j]*delta_o*s[j];
                                }
                                for(i=0;i<3;i++)
                                {
                                        s[i]=s[i]+LR*delta_o*y[i];
                                }
                                for(j=0;j<2;j++)
                                {
                                        for(i=0;i<4;i++)
                                        {
                                                w[j][i]=w[j][i]+LR*delta_h[j]*x[p][i];
                                        }
                }
                error=error+((t[p]-z)*(t[p]-z))/2;
        }
                error=error/11;
                printf("Iterasi: %d Error: %f\n",l,error);
                fprintf(f,"%f\n",error);
                if(error<0.00001)break;
}
fclose(f);
```

```
	//running
		x[0][0]=0.67;
		x[0][1]=0.67;
		x[0][2]=0.67;
		printf("IPK: %.2f\n",x[0][0]);
		printf("Psikologi: %.2f\n",x[0][1]);
		printf("Wawancara: %.2f\n",x[0][2]);
		for(j=0;j<2;j++)
		{
			O[j]=0.0;
			for(i=0;i<4;i++)
			{
				O[j]=O[j]+x[0][i]*w[j][i];
			}
			y[j]=sigmoid(O[j]);
		}
		O[0]=0.0;
		for(i=0;i<3;i++)
		{
			O[0]=O[0]+y[i]*s[i];
		}
		z=sigmoid(O[0]);
		printf("Output:%.2f\n",z);
		if(z<0.5)
			printf("Keputusan: TIDAK LULUS\n");
		else
			printf("Keputusan: LULUS\n");
		getch();
		return 0;
}
```

Hasil Program