Praktikum Sistem Cerdas

NRP	:	3223600019
Nama	:	Muhammad Bimo Fachrizky
Materi	:	Membuat Program Aplikasi Backpropagation
Tanggal	:	Senin, 17 Maret 2025

Praktikum 5

Membuat Program Aplikasi Backpropagation

I. Tujuan Pembelajaran

- Mahasiswa dapat memahami dan menjelaskan konsep Backpropagation
- Mahasiswa dapat menjelaskan model Backpropagation
- Mahasiswa dapat membuat aplikasi Backpropagation

Software yang di perlukan

- Microsoft Visual C++
- PyCharm

II. Langkah percobaan

1. 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 main() {
   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},
```

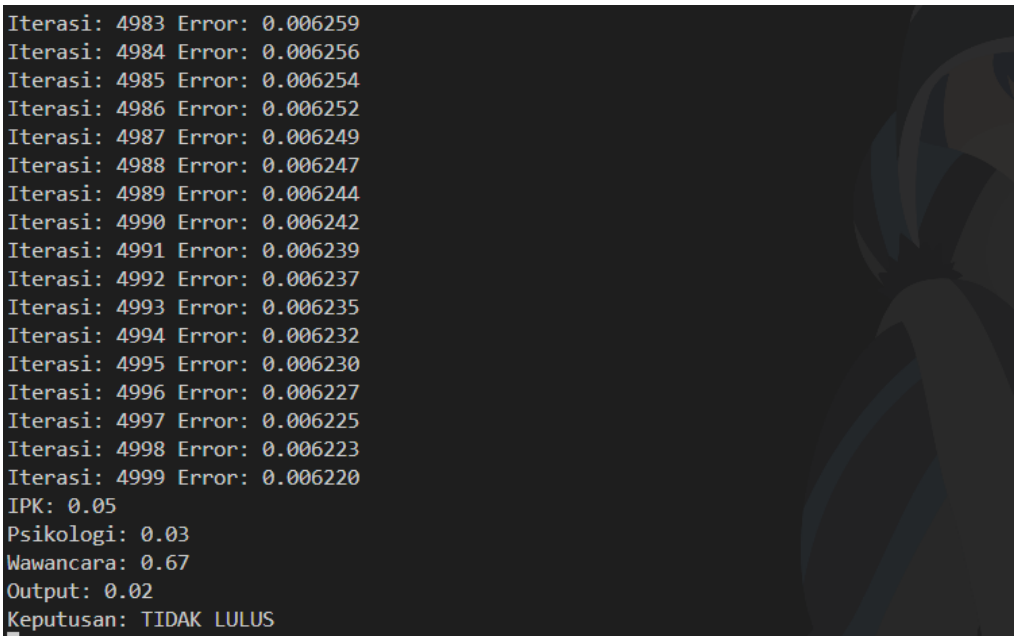
```
    {0.67, 0.33, 0.33, 1.00},
    {0.33, 1.00, 0.67, 1.00},
    {0.33, 0.67, 0.33, 1.00},
    {0.33, 0.33, 0.67, 1.00}
};
float t[11] = {1.0, 1.0, 1.0, 0.0, 1.0, 1.0, 1.0, 0.0, 1.0, 0.0, 1.0};
float w[6][4], O[6], s[6], LR = 0.1f, init = 0.15f, error;
FILE *f;
f = fopen("error.txt", "w");

// 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] += x[p][i] * w[j][i];
            }
            y[j] = sigmoid(O[j]);
        }
        O[0] = 0.0;
        for (i = 0; i < 3; i++) {
            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++) {
```

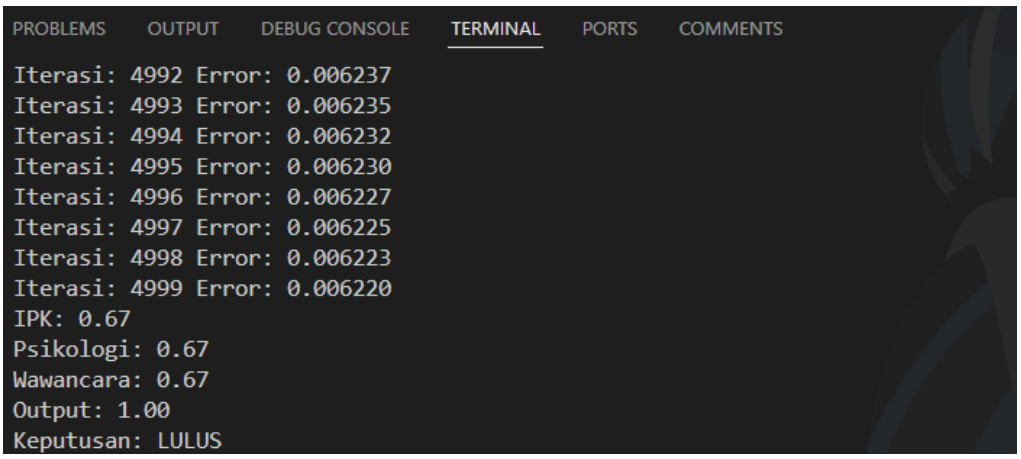
```
            delta_h[j] = f1[j] * delta_o * s[j];
        }
        for (i = 0; i < 3; i++) {
            s[i] += LR * delta_o * y[i];
        }
        for (j = 0; j < 2; j++) {
            for (i = 0; i < 4; i++) {
                w[j][i] += LR * delta_h[j] * x[p][i];
            }
        }
        error += ((t[p] - z) * (t[p] - z)) / 2;
    }
    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] += x[0][i] * w[j][i];
    }
    y[j] = sigmoid(O[j]);
}
O[0] = 0.0;
for (i = 0; i < 3; i++) {
    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;
}
```

III. Hasil Percobaan

- Multiperceptron





IV. Analisa

Praktikum di atas melakukan sebuah percobaan mengimplementasikan algoritma backpropagation dalam jaringan saraf tiruan untuk menentukan kelulusan berdasarkan beberapa parameter seperti IPK, hasil tes psikologi, dan wawancara. Program menggunakan arsitektur Multilayer Perceptron dengan satu lapisan tersembunyi yang terdiri dari enam neuron dan satu neuron keluaran. Bobot awal diinisialisasi secara acak dalam rentang tertentu menggunakan fungsi d_rand(), dan fungsi aktivasi sigmoid digunakan untuk menormalisasi keluaran dari setiap neuron.

Selama proses pelatihan, program melakukan propagasi maju untuk menghitung keluaran dari jaringan berdasarkan bobot yang ada, kemudian menghitung kesalahan antara hasil keluaran dan target yang diinginkan. Kesalahan ini digunakan dalam propagasi balik (backpropagation) untuk memperbarui bobot dengan metode gradient descent, sehingga jaringan dapat menyesuaikan diri dengan pola data yang diberikan. Pelatihan berlangsung hingga kesalahan mencapai ambang batas tertentu atau jumlah iterasi maksimal tercapai. Nilai error dicetak pada setiap iterasi dan juga disimpan dalam file "error.txt" untuk analisis lebih lanjut.

Setelah pelatihan selesai, program masuk ke tahap pengujian, di mana pengguna dapat memasukkan data baru untuk melihat apakah seseorang lulus atau tidak berdasarkan hasil prediksi jaringan. Program akan menghitung keluaran menggunakan bobot yang telah diperbarui, dan jika nilai keluaran lebih dari 0.5, maka keputusan yang diambil adalah "LULUS"; jika kurang dari 0.5, keputusan adalah "TIDAK LULUS". Secara keseluruhan, program ini menunjukkan bagaimana jaringan saraf tiruan dapat digunakan dalam pengambilan keputusan berbasis data dengan pendekatan pembelajaran mesin yang efektif.

V. Kesimpulan

Program ini berhasil mengimplementasikan algoritma backpropagation dalam jaringan saraf tiruan untuk memprediksi kelulusan berdasarkan beberapa parameter input. Dengan menggunakan metode gradient descent, bobot dalam jaringan diperbarui secara bertahap hingga kesalahan mencapai ambang batas yang diinginkan. Fungsi aktivasi sigmoid memungkinkan jaringan untuk menangani hubungan non-linear dalam data, sehingga mampu melakukan klasifikasi dengan lebih akurat. Hasil dari pelatihan menunjukkan bahwa jaringan dapat mengenali pola dengan baik dan memberikan prediksi yang sesuai.