BAB II

TINJAUAN PUSTAKA

A. Penelitian Terdahulu

Berikut beberapa penelitian terdahulu yang terkait dengan metode *backpropagation* diantaranya yaitu:

1. Nurdiansyah et al. (2019) menggunakan metode *backpropagation* untuk mengklasifikasi ketepatan waktu kelulusan mahasiswa. Variabel data yang digunakan adalah 9 input, yaitu nilai Insdeks Prestasi (IP) semester 1 sampai 6, jumlah SKS yang ditempuh, semester saat terakhir kali memprogram matakuliah Kuliah Kerja Nyata (KKN), dan Praktek Kerja Lapangan (PKL). Penerapan metode klasifikasi ini dilakukan dengan menggunakan *learning rate* 0.1, 0.3, 0.5, 0.7, dan 0.9 dengan batas iterasi masing-masing 1000, 2000, dan 3000. Hasil pengujian diperoleh bahwa nilai akurasi tertinggi yaitu sebesar 98.82% pada iterasi ke-2000 dan 3000, masing-masing dengan *learning rate* = 0.7 dan 0.9 untuk iterasi ke-2000 dan *learning rate* = 0.5, 0.7 dan 0.9 untuk iterasi ke-3000.
2. Prasetyawan et al. (2018) menerapkan jaringan syaraf tiruan *backpropagation* untuk mengklasifikasi masa studi sarjana. Arsitektur yang digunakan sebanyak 7, yaitu hasil tes penerimaan siswa, jurusan, Indeks Prestasi Semester (IPS) 1, Indeks Prestasi Semester (IPS) 2, jenis kelamin, jenis sekolah menengah, dan lama

belajar. Hasil penelitian untuk klasifikasi masa studi sarjana dengan tingkat akurasi di atas 85%.

3. Yalidhan (2018) melakukan sebuah penelitian menggunakan algoritma *backpropagation* untuk memprediksi kelulusan mahasiswa. Variabel yang digunakan terdiri dari data gender, Indeks Prestasi Semester (IPS) 1, jumlah pengambilan Satuan Kredit Semester (SKS) 2, Indeks Prestasi Semester (IPS) 2, jumlah pengambilan Satuan Kredit Semetser (SKS) 3, Indeks Prestasi Semester (IPS) 3, jumlah pengambilan Satuan Kredit Semester (SKS) 4, Indeks Prestasi Semester (IPS) 4, jumlah pengambilan Satuan Kredit Semester (SKS) 5, dan Indeks Prestasi Semester (IPS) 5. Penelitian ini menggunakan sebanyak 318 sampel data yang mana 70% data yang digunakan sebagai data trainning dan data 30% data digunakan sebagai data testing. Hasil penelitian menunjukkan persentase akurasi prediksi sebesar 98.97%.

B. Landasan Teori

1. Klasifikasi

Klasifikasi adalah proses menemukan sekumpulan model atau fungsi yang menjelaskan dan membedakan data ke dalam kelas-kelas tertentu, denga tujuan menggunakan model tersebut dalam menentukan kelas dari suatu objek yang belum diketahui kelasnya (Michael et al., 2000). Ada 2 proses dalam klasifikai, yaitu:

a. Proses *learning/training*

Melalukan pembangunan model menggunakan data *training*.

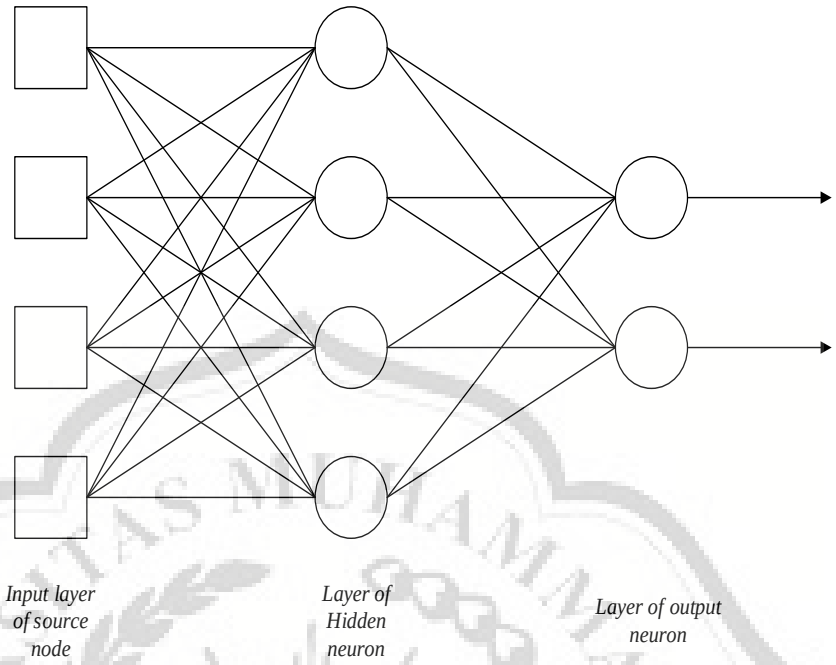
b. Proses *testing*

Melakukan tes terhadap data *testing* meggunakan model yag telah diperoleh dari proses *training.*

2. Model Neuron

Sel syaraf *neuron* adalah unit pemrosesan informasi yang merupakan dasar dari operasi Jaringan Syaraf Tiruan yang disusun berhubungan erat dengan algoritma belajar yang digunakan untuk melatih jaringan. Arsitektur JST dapat dibedakan menjadi empat, yaitu: Single-layer *Feedforword Nework, Multi-layer Feedforword Nework, Recurrent Network,* dan *Lattice Structure* (Suyanto, 2014).

Dalam penelitian ini, arsitektur jaringan yang digunkan adalah *Multi-layer Feedforword Nework* (jaringan dengan banyak lapisan). Kelas ke dua dari *Feedforword Neural Nework* adalah jaringan dengan satu atau lebih lapisan tersembunyi *(lapisan tersembunyi)*, dengan *computation nodes* yang berhubungan disebut *hidden neurons* atau *hidden units.* Gambar 1 mengilusrasikan *Multi-layer Feedforword Neural Nework* untuk kasus satu *lapisan tersembunyi* . Untuk menyingkat jaringan pada gambar 2 tersebut, biasanya disebut sebagai jaringan 4-4-2 (Suyanto, 2014).



Gambar 1. Jaringan *feedforward* (4-4-2).

3. *Backpropagation*

Backpropagation merupakan algoritma pembelajaran yang terawasi dan biasanya digunakan oleh *perceptron* dengan banyak lapisan untuk mengubah bobot-bobot yang terhubung dengan neuron yang ada pada lapisan tersembunyi (Kusumadewi, 2004).

Algoritma *backpropagation* menggunakan *error output* untuk mengubah nilai bobotnya dalam arah mundur (*backward*). Untuk mendapatkan error ini tahap perambatan maju (*forward propagation*) harus dikerjakan terlebih dahulu (Kusumadewi, 2004).

4. Pelatihan *Backpropagation*

Menurut Cyntia, E.P. & Ismanto (2017) pelatihan *backpropagation* dilakukan melalui langkah-langkah berikut ini:

Langkah 0 : Inisialisasi semua bobot dengan bilangan acak kecil;

Langkah 1 : Selama kondisi berhenti bernilai salah, kerjakan langkah 2 – 9.

Langkah 2 : Untuk setiap data training, lakukan langkah 3 – 8.

Langkah 3 : Untuk langkah 3 hingga 5 merupakan Proses Umpan Maju (Feedforward). Setiap unit input (X_i , i = 1,…, n); menerima sinyal input dan menyebarkan sinyal tersebut ke seluruh unit tersembunyi.

Langkah 4 : Pada setiap unit tersembunyi (Z_j , j = 1,…, p); menjumlahkan sinyal-sinyal input yang sudah berbobot (termasuk bias nya) dengan rumus yang ditunjukkan oleh persamaan 1 sebagai berikut,

$$Z\_in_j = V0_j + \sum_{i=1}^{n} X_i \, V_{ij} \qquad (1)$$

Lalu menghitung sinyal output dari unit tersembunyi dengan menggunakan fungsi aktivasi yang telah ditentukan.

$$Z_j = f\,(Z_{inj}) \qquad (2)$$

Sinyal output ini selanjutnya dikirim ke seluruh unit pada unit diatas (unit output).

Langkah 5 : Tiap-tiap output (Y_k, k = 1,…, m); menjumlahkan bobot sinyal input

$$y\_in_k = w0_k + \sum_{i=1}^{n} z_i \, w_{ij} \qquad (3)$$

Lalu menghitung sinyal output dari unit output bersangkutan dengan menggunakan fungsi aktivasi

yang telah ditentukan

$$y_k = f\,(y_{ink}) \qquad (4)$$

Sinyal output ini selanjutnya dikirim ke seluruh unit output.

Langkah 6 : Untuk langkah 6 hingga 7 merupakan Proses Umpan Mundur (Backrward) / Propagasi *Error*. Setiap unit output (Y_k, k = 1,…, m); menerima suatu pola target yang sesuai dengan pola input pelatihan, untuk menghitung kesalahan *(error)* antara target dengan output yang dihasilkan jaringan;

$$\delta_j = t_{k-} y_y \;\; f'(y_{ink}) \qquad (5)$$

Faktor δ_k digunakan untuk menghitung koreksi *error* (Δw_{jk}) yang nantinya akan dipakai untuk memperbaiki w_k dimana :

$$\Delta w_{ok} = \alpha \delta_k z_j \qquad (6)$$

Selain itu juga dihitung koreksi bias (Δw_{ok}) yang nantinya akan dipakai untuk memperbaiki w_{ok} dimana:

$$\Delta w_{ok} = \alpha \delta_k \qquad (7)$$

Faktor δ_k kemudian dikirimkan ke lapisan yang berada pada langkah 7.

Langkah 7 : Setiap unit tersembunyi (Z_j , j = 1,…, p); menerima

input delta (dari langkah ke- 6) yang sudah berbobot :

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k \, w_{jk} \quad (8)$$

Kemudian hasilnya dikalikan dengan turunan dari fungsi aktivasi yang digunakan untuk menghitung informasi kesalahan *error* δ_i dimana :

$$\delta_j = \delta\_in_j \, f'(Z_{ink}) \quad (9)$$

Kemudian hitunglah koreksi bobot dengan :

$$\Delta v_{ij} = \alpha \delta_j \, x_i \quad (10)$$

Kemudian hitunglah koreksi bias :

$$\Delta v_{oj} = \alpha \delta_j \quad (11)$$

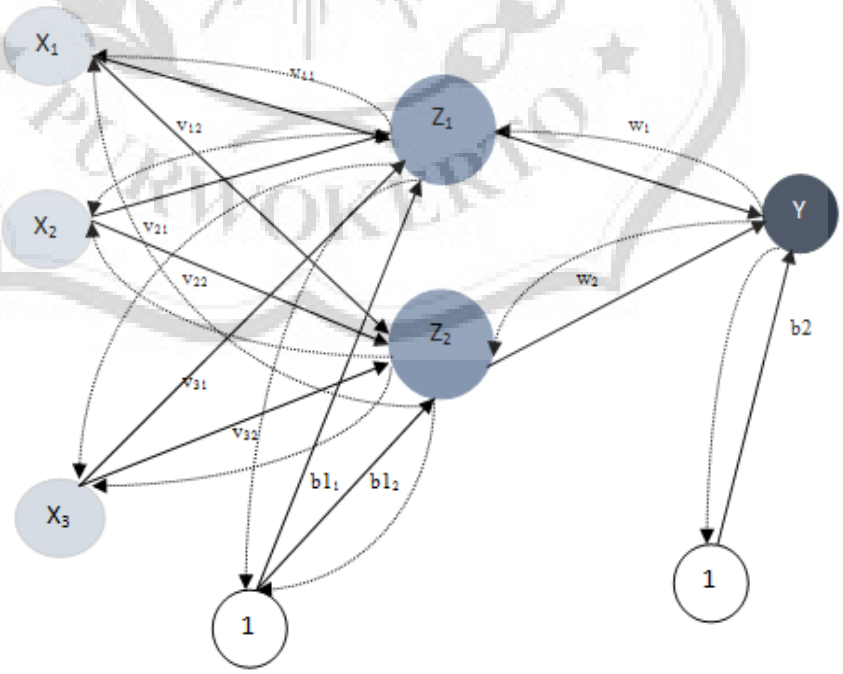
Langkah 8 : Untuk langkah 6 hingga 7 merupakan Proses Update Bobot dan Bias. Setiap unit *output* (Y_k, k = 1,…, m); memperbaiki bobot dan bias dari setiap unit tersembunyi (j = 0,…, p);

$$w_{jk} \; baru \; = w_{jk} \; lama \; + \; w_{jk} \quad (12)$$

Langkah 9 : Tes kondisi berhenti apabila error ditemukan. Jika kondisi berhenti terpenuhi, maka pelatihan jaringan dapat dihentikan.

5. Arsitektur *Backpropagation*

Arsitektur jaringan *backpropagation* seperti terlihat pada Gambar 2 Jaringan terdiri atas 3 unit masukan ($\boldsymbol{x_1}$, $\boldsymbol{x_2}$ dan $\boldsymbol{x_3}$), 1 lapisan tersembunyi dengan 2 *neuron* ($\boldsymbol{z_1}$ dan $\boldsymbol{z_2}$), dan 1 unit lapisan keluaran (*Y*). Bobot yang menghubungkan $\boldsymbol{x_1}$, $\boldsymbol{x_2}$ dan $\boldsymbol{x_3}$ dengan *neuron* pertama pada lapisan tersembunyi adalah $\boldsymbol{v_{11}, v_{21}, \text{dan } v_{31}}$, sedangkan bobot yang menghubungkan $\boldsymbol{x_1}$, $\boldsymbol{x_2}$ dan $\boldsymbol{x_3}$ dengan *neuron* kedua pada lapisan tersembunyi adalah $\boldsymbol{v_{12}, v_{22}, \text{dan } v_{32}}$. $\boldsymbol{b1_1 \text{ dan } b1_2}$ adalah bobot bias yang menuju ke *neuron* pertama dan kedua pada lapisan tersembunyi. Bobot yang menghubungkan $\boldsymbol{z_1}$ dan $\boldsymbol{z_2}$ dengan *neuron* pada lapisan keluaran adalah $\boldsymbol{w_1 \text{ dan } w_2}$. Bobot bias *b2* menghubungkan lapisan tersembunyi dengan lapisan keluaran (Kusumadewi, 2004).



Gambar 2. Arsitektur Jaringan *Backpropagation*

6. Algoritma Pelatihan

Ada beberapa algoritma pelatihan yang terdapat dalam jaringan syaraf tiruan, diantaranya adalah *Fletcher-Reeves Update (traincgf), Polak-Ribiere (traincgp), Powell-Beale Restart (traincgb), Scaled Conjugate Gradient (trainscg), Gradient Descent (traingd), Gradient Descent* dengan *Adaptive Learning Rate (traingda), Gradient Descent* dengan *Momentum (traingdm), Gradient Descent* dengan *Momentum* dan *Adaptive Learning Rate (traingdx), Resilent Backpropagation (trainrp), BFGS (trainbfg), One Step Secant (trainoss), Levenberg-Marquardth (trainlm)* (Kusumadewi, 2004)*)*.

Pada penelitian sebelumnya, Mustafidah & Suwarsito (2015) telah melakukan penelitian dengan pengujian terhadap 12 algoritma pelatihan yang terdapat dalam jaringan *backpropagation*. Pengujian dilakukan menggunakan parameter jaringan seperti target error = 0.001 (10-3), maksimum epoh = 10000, learning rate (lr) = 0.01, dengan 5 neuron input dan satu neuron output. Berdasarkan hasil pengujian dari 12 algoritma pelatihan, algoritma *trainlm* memiliki error terkecil dengan level $\alpha = 5\%$ dan memberikan error 0,0001986858.

7. Algoritma *Levenberg Marquardt*

Algoritma *levenberg marquardt* dikembangkan oleh Kenneth dan Donald Marquardt, memberikan solusi numerik untuk masalah meminimalkan fungsi nonlinier. Dalam bidang jaringan syaraf tiruan, algoritma ini cocok untuk pelatihan kecil dan menengah.

Algoritma *levenberg marquardt* merupakan pengembangan dari algoritma *backpropagation.* Algoritma ini dibangun untuk mengatasi beberapa kekurangan yang ada pada algoritma *backpropagation* dengan memanfaatkan teknik optimasi numerik standar. Tujuan dari *levenberg marquardt* adalah meminimalkan total *error* (Arhami & Desiani, 2005).

8. MSE *(Mean Square Error)*

Perhitungan kesalahan atau *error* merupakan pengukuran bagaimana jaringan dapat belajar dengan baik sehingga jika dibandingkan dengan pola yang baru akan dengan mudah dikenali. Kesalahan pada keluaran jaringan merupakan selisih antara keluaran sebenarnya (*current output)* dan keluaran yang diinginkan *(desired output)*. Selisih yang dihasilkan antara keduanya biasanya ditentukan dengan cara dihitung menggunakan suatu persamaan. Dalam penelitian ini, *error* (kesalahan) dihitung menggunakan MSE (*Mean Squared Error)*. MSE merupakan fungsi kinerja jaringan yang mengukur kinerja berdasarkan rata-rata dari kuadrat *error*nya (Hartati & Kusumadewi, 2006).

9. Lapisan Tersembunyi

Menentukan lapisan tersembunyi pada jaringan syaraf tiruan sangatlah penting, jumlah neuron lapisan tersembunyi yang digunakan berpengaruh besar terhadap perubahan hasil dari pelatihan jaringan tersebut. Jika jumlah neuron lapisan tersembunyi terlalu sedikit, data

yang akan dihasilkan oleh jaringan sering kali tidak relevan. Namun jika neuron terlalu banyak, akan memperlambat proses pelatihan dari jaringan dan akan terjadi pelatihan yang tidak hingga. Maka dari itu, diperlukanlah pencocokan terhadap jumlah neuron lapisan tersembunyi yang digunakan. Sangat banyak aturan yang digunakan untuk menentukan jumlah neuron lapisan tersembunyi yang tepat untuk setiap jaringan (Heaton, 2008). Beberapa aturan yang sering dipakai adalah:

a. Jumlah neuron lapisan tersembunyi lebih besar dari jumlah neuron input atau jumlah neuron output.
b. Jumlah neuron lapisan tersembunyi seharusnya 2/3 besar dari jumlah neuron input ditambah jumlah neuron output.
c. Jumlah neuron lapisan tersembunyi harus lebih kecil atau sama dengan dua kali jumlah input layer. Fungsi Aktivasi

10. Fungsi Aktivasi

Fungsi aktivasi merupakan bagian penting dalam tahapan perhitungan keluaran dari suatu algoritma. Fungsi aktivasi yang digunakan adalah fungsi sigmoid biner. Pada umumnya fungsi sigmoid biner digunakan untuk Jaringan Syaraf Tiruan (JST) yang dilatih dengan menggunakan metode *backpropagation* yang memiliki nilai antara 0 sampai 1 (Suyanto, 2014). Rumus fungsi sigmoid biner pada persamaan berikut:

$$f(x) = \frac{1}{1+e^{-x}}$$ dengan turunan: $f\ x\ = f\ x\ (1 - f\ x\ )$ (13)

11. Matlab

Matlab adalah perangkat lunak yang digunakan untuk pemrograman analisis, sesrta komputasi teknik dan matematis berbasis matriks. Matlab mengintegrasikan komputasi matematik, visualisasi, dan bahasa pemograman untuk memberikan lingkungan fleksibel bagi komputasi teknis. Arsitektur terbukanya memudahkan pengguna dalam mengeksplorasi data, menciptakan algoritma, dan menciptakan beberapa perangkat grafik (GUI). Matlab dapat digunakan untuk komputasi numerik dan pengembangan algoritma serta analisis data dan pemrosesan sinyal (Sianipar, 2015).

Matlab dapat dipakai secara interaktif dan memiliki fungsi-fungsi yang sangat memudahkan pekerjaan pemrograman. Dalam aspek komputasi, matlab merupakan perangkat lunak yang sangat tangguh yang terlibat dalam permasalahan-permasalahan sains dan keteknikan. Matlab melibatkan beberapa toolbox seperti *Symbolic Math Toolbox*, *Control System Toolbox*, dan *Signal Processing Toolbox* dalam perancangan dan analisisnya (Sianipar, 2015).