BAB II

KAJIAN PUSTAKA

A. Penelitian Terdahulu

Pada penelitian sebelumnya dilakukan oleh Hizham et al. (2018) untuk mengklasifikasi ketepatan waktu kelulusan mahasiswa dengan metode jaringan syaraf tiruan *Backpropagation*. Data yang digunakan yaitu data lulusan mahasiswa Program Studi Sistem Informasi Universitas Jember angkatan tahun 2011-2013. Atribut yang digunakan untuk klasifikasi berjumlah 9 atribut. Nilai akurasi tertinggi yaitu sebesar 98,82% pada iterasi ke-2000 dan 3000 dengan *learning rate* 0,7 dan 0,9. Hasil tersebut didapat dari jumlah data benar sebanyak 167 data dari 169 data secara keseluruhan.

Dalam penelitian yang dilakukan oleh Wongkhamdi & Seresangtakul (2010) telah dilakukan penelitian perbandingan antara analisis diskriminan klasik dan jaringan syaraf tiruan dalam memprediksi hasil kelulusan mahasiswa. Data yang digunakan yaitu data mahasiswa program studi ilmu komputer angkatan tahun 2000-2004. Sebanyak 594 data digunakan untuk pelatihan dan pengujian. Variable yang digunakan sebanyak 10. Untuk membandingkan kinerja pendekatan jaringan dengan analisis diskriminan berdasarkan tingkat klasifikasi rata benar (Mean CCR). Hasil menunjukkan bahwa jaringan syaraf tiruan mampu meningkatkan kelulusan mahasiswa secara signifikan dibandingkan dengan analisis diskriminan. Akurasi terbaik

diperoleh jaringan syaraf tiruan sebesar 93,3% dan analisis diskriminan sebesar 81,5%.

Bahadir (2015) telah melakukan penelitian untuk memprediksi kinerja mahasiswa tingkat dua menggunakan jaringan syaraf tiruan di Fakultas Teknik dan teknologi informasi dengan metode *Backpropagation*. Jaringan syaraf yang digunakan yaitu jaringan *feed-Forward* dengan lapisan input 10, lapisan tersembunyi 6 dan 4 lapisan output. Total 150 mahasiswa tingkat dua digunakan dalam analisis. 60% dari total data digunakan sebagai data pelatihan, 30% sebagai data uji dan 10% digunakan untuk validasi silang. Hasil jaringan syaraf mampu memprediksi secara akurat 11 dari 13 untuk data yang memuaskan, 10 dari 12 untuk data yang sangat baik, 9 dari 11 untuk data yang baik dan 8 dari 9 untuk data yang buruk. Hasi keseluruhan diperoleh akurasi prediksi sebesar 84.6 %.

B. Landasan Teori

1. Prediksi

Prediksi adalah usaha untuk memperkirakan sesuatu yang akan terjadi di waktu mendatang dengan memanfaatkan berbagai informasi yang relevan pada waktu-waktu sebelumnya melalui suatu metode ilmiah. Prediksi juga memerkirakan besar atau jumlah sesuatu pada waktu yang akan datang berdasarkan data pada masa lampau yang dianalisis secara ilmiah (Hutabarat et al., 2018).

2. Kelulusan

Sesuai dengan Buku Panduan Akademik Fakultas Teknik Universitas Muhammadiyah Purwokerto tahun 2016. Mahasiswa wajib lulus minimal enam puluh satuan kredit semester dengan IPK minimal dua pada empat semester pertama.

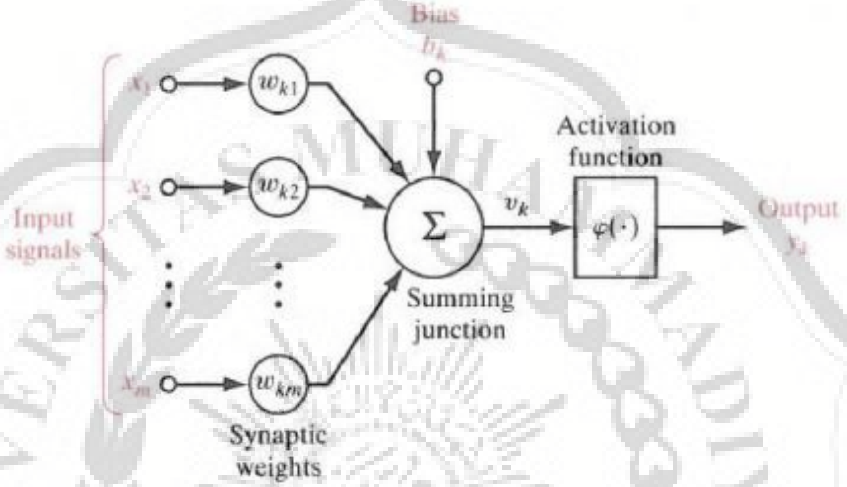
3. Jaringan Syaraf Tiruan

Jaringan syaraf tiruan adalah metode komputasi yang meniru jaringan syaraf biologis. Metode ini menggunakan perhitungan non-linear dasar yang disebut neuron dan saling berhubungan sehingga menyerupai jaringan syaraf manusia. Jaringan syaraf tiruan dibuat untuk memecahkan masalah pengenalan pola atau klasifikasi. Jaringan saraf tiruan tidak diprogram untuk menghasilkan keluaran tertentu karena didasarkan pengalamannya selama mengikuti proses pembelajaran. Pada proses pembelajaran, ke dalam jaringan saraf tiruan dimasukkan pola-pola masukan (dan keluaran) lalu jaringan akan diajari untuk memberikan jawaban yang bisa diterima (Puspitaningrum, 2006).

Prinisp jaringan saraf tiruan (JST) secara sederhana digambarkan pada Gambar 1. dan ditentukan oleh tiga elemen dasar model saraf, yaitu:

a. Satu set dari sinapsis, atau penghubung yang masing-masing digolongkan oleh bobot atau kekuatannya.
b. Sebuah penambah untuk menjumlahkan sinyal-sinyal input. Ditimbang dari kekuatan sinaptik masing-masing neuron.

c. Sebuah fungsi aktivasi untuk membatasi amplitude output dari neuron. Fungsi ini bertujuan membatasi jarak amplitude yang diperbolehkan oleh sinyal output menjadi sebuah angka yang terbatas.



Gambar 1. Prinsip Jaringan Syaraf Tiruan (Nurhani et al., 2018)

4. Konsep Dasar Jaringan Syaraf Tiruan

Nilai dari *input* dan *output* akan diproses di dalam neuron, dan neuron berkumpul di dalam lapisan yang disebut neuron *layers* (Lesnussa et al., 2017). Untuk lapisan neuron tersebut ada 3, yaitu:

a. *Input layer*, adalah lapisan yang berisi unit-unit masukan yang menggambarkan suatu permasalahan dari luar.

b. *Hidden layer*, adalah lapisan untuk memroses unit *input* sebelum menuju lapisan *output.*

c. *Output layer,* adalah lapisan yang berisi solusi atau hasil dari pemrosesan dari suatu permasalahan.

5. Arsitektur Jaringan Syaraf Tiruan

Terdapat tiga jenis arsitektur jaringan saraf tiruan menurut (Kusumadewi, 2003):

a. *Single layer network*, Jaringan yang hanya memiliki satu lapisan yang terdiri dari 1 *input layer* dan 1 *output layer*. Seluruh unit yang berada di dalam *input layer* akan selalu terhubung dengan *output layer*. Proses yang dilakukan dengan arsitektur ini adalah secara langsung mengolah *input* untuk kemudian menjadi *output*. Beberapa contoh algoritma yang menggunakan arsitektur ini adalah *Perceptron, Hopfield,* dan *ADALINE.*

b. *Multi layer network,* Arsitektur ini memiliki 3 lapisan yaitu *input layer*, *hidden layer*, dan *output layer*. Arsitektur ini mampu menyelesaikan masalah yang lebih rumit jika dibanding dengan *single layer network*, tetapi biasanya dengan konsekuensi proses yang lebih lama. Beberapa contoh algoritma yang menggunakan arsitektur ini adalah *backpropagation, neocognitron,* dan *MADALINE*.

c. *Competitive layer network,* Sekumpulan *neuron* bersaing untuk mendapatkan hak menjadi aktif di dalam jaringan ini. Contoh algoritma yang menggunakan arsitektur ini yaitu LVQ.

6. Metode Pelatihan Jaringan Saraf Tiruan

Terdapat tiga metode dari pelatihan jaringan saraf tiruan menurut (Puspitaningrum, 2006):

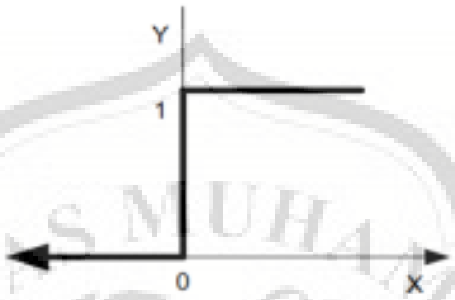
a. *Supervised Learning*, selisih (*error*) dari pola *output* aktual dengan *output* target (yang diinginkan) digunakan untuk mengubah bobot dengan tujuan untuk menghasilkan *output* aktual yang sedekat mungkin dengan *output* target. Contoh algoritma yang menggunakan metode ini adalah *Backpropagation, ADALINE,* dan *Perceptron.*
b. *Unsupervised Learning*, dengan metode ini kita tidak dapat menentukan *output* target, kita hanya bisa menyusun nilai bobot dalam *range* tertentu tergantung dari *input* yang diberikan. Tujuan dari metode ini mengelompokkan unit-unit yang hampir sama dalam suatu area tertentu. Contoh algoritma menggunakan metode ini adalah: *Neocognitron, LVQ,* dan *Competitive.*
c. *Hybrid Learning*, metode penggabungan dari dua metode di atas, penentuan bobot sebagian dilakukan melalui *supervised learning* dan sebagian menggunakan *unsupervised learning.*

7. Fungsi Aktivasi

Fungsi aktivasi berperan dalam menentukan *output* dari suatu neuron (Kusumadewi, 2004). Ada tiga fungsi aktivasi yang umum dipakai yaitu:

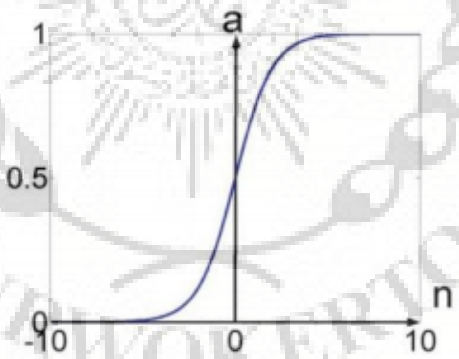
a. Fungsi *threshold,* yaitu fungsi *threshold* biner. Untuk kasus yang memiliki bilangan bipolar, maka angka 0 diganti dengan angka -1. Dalam jaringan saraf tiruan terkadang ditambahkan suatu unit *input* yang selalu bernilai 1.

Unit tersebut disebut dengan bias. Bias berfungsi untuk mengubah *threshold* menjadi = 0.



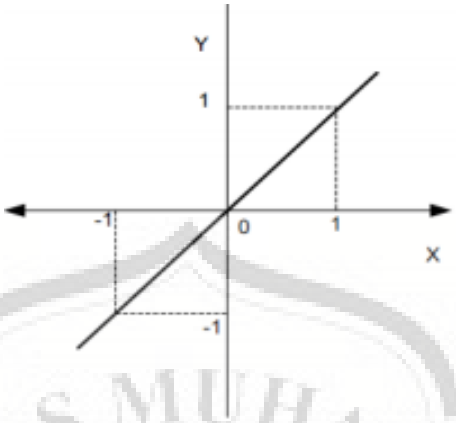
Gambar 2. Fungsi aktivasi *Threshold* (Kusumadewi, 2004)

b. Fungsi *Sigmoid*, fungsi ini cukup sering digunakan karena kemudahannya dalam mendiferensiasikan nilai fungsinya.



Gambar 3. Fungsi aktivasi *Sigmoid* (Kusumadewi, 2004)

c. Fungsi Identitas, fungsi ini dipakai ketika *output* yang dihasilkan oleh jaringan saraf tiruan merupakan sembarang bilangan riil (bukan hanya pada *range* [0,1] atau [1,- 1]). X=Y.



Gambar 4. Fungsi aktivasi Identitas (Kusumadewi, 2004)

8. Metode *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. 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. Pada saat perambatan maju, *neuron-neuron* diaktifkan dengan fungsi aktivasi yang dapat didiferensiasikan. Informasi yang diberikan pada jaringan syaraf akan dirambatkan mulai dari lapisan *input* sampai ke lapisan *output* melalui lapisan yang lainnya, yang sering dikenal dengan nama lapisan tersembunyi atau *hidden layer* (Kusumadewi, 2004).

9. Algoritma Pelatihan *Backpropagation*

Pelatihan Backpropagation dilakukan melalui langkah-langkah berikut ini (Anike et al., 2012):

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).

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

Lalu menghitung sinyal output dari unit tersembunyi dengan menggunakan fungsi aktivasi yang telah ditentukan $Z_j = f\,(Z_{inj})$. 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:

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

Lalu menghitung sinyal output dari unit output bersangkutan dengan menggunakan fungsi aktivasi yang telah ditentukan $y_k = f\,(y_{ink})$. Sinyal output ini selanjutnya dikirim ke seluruh unit output.

Langkah 6 : Untuk langkah 6 hingga 7 merupakan proses umpan mundur (*Backrward*) atau 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;

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

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

w_k dimana :

$$\Delta w_{ok} = \alpha \delta_k z_j$$

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

$$\Delta w_{ok} = \alpha \delta_k$$

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 :

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

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

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

Kemudian hitunglah koreksi bobot dengan :

$$\Delta v_{ij} = \alpha \delta_j \, x_i$$

Kemudian hitunglah koreksi bias :

$$\Delta v_{oj} = \alpha \delta_j$$

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 seti ap unit tersembunyi (j = 0,…, p);

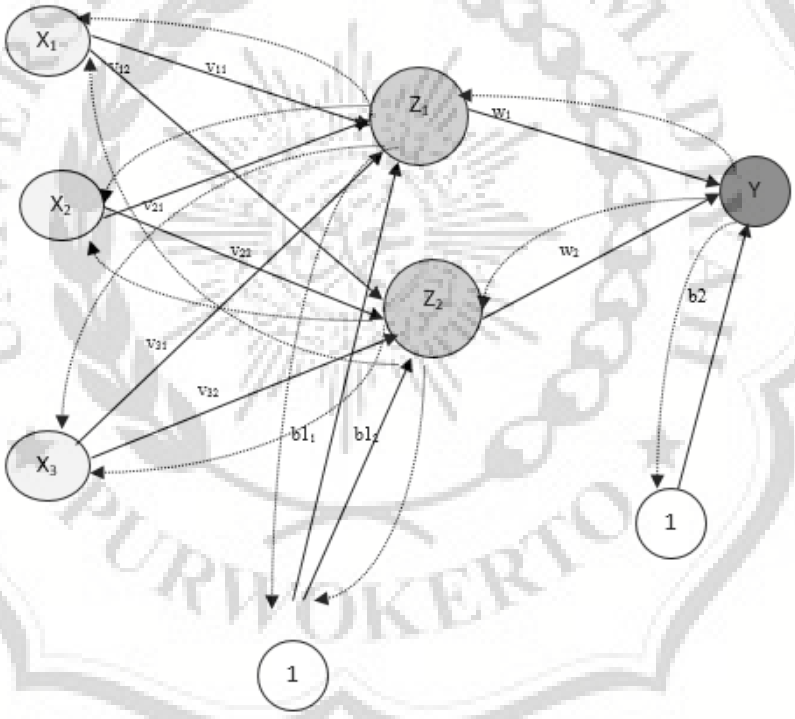
$$w_{jk}\ baru\ = w_{jk}\ lama\ + w_{jk}$$

Langkah 9 : Tes kondisi berhenti jika *error* ditemukan. Berhenti terpenuhi, maka pelatihan jaringan dapat dihentikan.

10. Arsitektur *Backpropagation*

Arsitektur jaringan *backpropagation* seperti terlihat pada Gambar 5. Jaringan terdiri atas 3 unit (*neuron*) pada lapisan *input*, yaitu x_1, x_2 dan x_3, 1 lapisan *output*, yaitu y. Bobot yang meghubungkan x_1, x_2 dan x_3 dengan neuron pertama pada lapisan tersembunyi adalah v_{11}, v_{12}, v_{13} (v_{ij}: bobot yang menghubungkan neuron *input* ke-i ke neuron ke-j pada lapisan tersembunyi). Perlu diingat bahwa, untuk pemakaian *toolbox* nnet pada Matlab, bobot v_{ij} memiliki pengertian yang sebaliknya (v_{ij}: bobot yang menghubungkan neuron ke-j pada suatu lapisan neuron ke-i pada lapisan sesudahnya). Misal: v_{12} adalah bobot yang menghubungkan neuron ke-2 pada

lapisan *input*, ke neuron ke-1 pada lapisan tersembunyi. Kembali ke Gambar 2.5, b_1 dan b_1 adalah bobot bias yang menuju ke neuron pertama dan kedua pada lapisan tersembunyi. Bobot yang menghubungkan z_1 dan z_2 dengan neuron pada lapisan *output*, adalah w_1 dan w_2. Bobot bias b_2 menghubungkan lapisan tersembunyi dengan lapisan *output*. Fungsi aktivasi yang digunakan, antara lapisan *input* dan lapisan tersembunyi, dan antara lapisan tersembunyi dengan lapisan *output* adalah fungsi aktivasi *logsig* (Kusumadewi, 2004).



Gambar 5. Arsitektur *Backpropagation*

11. 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 pengujian terhadap 12 algoritma pelatihan yang terdapat dalam jaringan *backpropagation*. Pengujian dilakukan untuk mendapatkan algoritma pelatihan yang paling optimal ditinjau dari *error* yang dihasilkan. Data masukan jaringan digunakan data random dengan 5 neuron input. Diperoleh hasil bahwa algoritma pelatihan *trainlm* memiliki *error* terkecil dibandingkan dengan 11 algoritma pelatihan lainnya yaitu dengan rata-rata *error* sebesar 0,0002196 dengan kontrol parameter jaringan target *error* = 0.001, maksimum epoh = 10000 dan *learning rate* (lr) = 0.05.

12. Lapisan Tersembunyi

Menentukan jumlah neuron di lapisan tersembunyi adalah bagian yang penting dalam menentukan arsitektur *neurol network,* walaupun dalam lapisan ini tidak secara langsung beinteraksi dengan lingkungan eksternal akan tetapi, lapisan ini memiliki pengaruh besar pada hasil akhir. Lapisan tersembunyi seharusnya kurang dari dua kali neuron lapisan input. Pada jumlah neuron

yang tersembunyi diantara 2/3 ukuran lapisan input ditambah neuron lapisan keluaran (Heaton, 2008).

13. 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. Persamaan yang digunakan adalah seperti pada persamaan 2.1 berikut (Kusumadewi & Hartati, 2006).

$$\text{MSE} = \frac{\sum p \sum j (T_{jp} - X_{jp})^2}{n_p n_o} \quad \text{.....(2.1)}$$

T_{jp} = nilai keluaran jaringan syaraf

X_{jp} = nilai target yang diinginkan untuk setiap keluaran

n_p = jumlah seluruh pola

n_o = jumlah keluaran

14. 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).