BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terkait

Pada penelitian yang berjudul "*Modified Backpropagation with Added White Gaussian Noise in Weighted Sum for Convergence Improvement*" yang dilakukan oleh (Sapkal dan Kulkarni 2018) menggunakan metode *Noise Injection* dengan tambahan *White Gaussian Noise* pada *8 Signal to noise* pada 2 bit *iris dataset (iris setosa, iris versicolour, iris virginica)* menghasilkan perbandingan waktu konvergensi sebagai berikut:

1. *BP* = 6050.
2. *NNBP* = **602.5.**

Penelitian ini menunjukkan bahwa *Backpropagation* dengan tambahan *Noise Net* manghasilkan waktu konvergensi lebih sedikit.

Pada penelitian yang berjudul "*AG-SGD : Angle-Based Stochastic Gradient Descent*" yang dilakukan oleh (Song, Pons, dan Yen 2021) menggunakan metode *AG-SGD* pada Data Sudut 0 – 180 derajat menghasilkan perbandingan nilai rata – rata *error* sebagai berikut:

1. *Vanilla* = 20.00.
2. *Momentum* = 19.92.
3. *RMSProp* = 17.48.
4. *Adam* = 18.93.
5. *Nadam* = 18.40.

6. *AdaMax* = 23.96.
7. *AdaDelta* = 14.62.
8. *AdaGrad* = 24.34.
9. *AMSGrad* = 17.73.
10. *AG-SGD* = **13.71.**

Penelitian ini menunjukkan bahwa *AG-SGD* menghasilkan nilai rata – rata *error* terendah.

Pada penelitian yang berjudul "*Plate Recognition Using Backpropagation Neural Network and Genetic Algorithm*" yang dilakukan oleh (Tarigan dkk. 2017) menggunakan metode *Genetic Algorithm* pada 220 Foto Data Plat Nomor Kendaraan menghasilkan perbandingan *epoch* dan *training time* sebagai berikut:

1. *Genetic Algorithm Backpropagation Neural Network (GABPNN):*
   a. *Epoch* = **2230.**
   b. *Training Time* = **3 Menit 9 Detik.**

2. *Backpropagation Neural Network (BPNN):*
   a. Epoch = 3530.
   b. *Training Time* = 3 Menit 43 Detik.

Penelitian ini menunjukkan bahwa *GABPNN* menghasilkan *epoch* dan *training time* lebih sedikit.

Pada penelitian yang berjudul "*Identifying Colorectal Tumor For Single Cell RNA Sequence Using Rectified Linear Unit With Stochastic Gradient Descent*" yang dilakukan oleh (Rajesh dkk. 2023) menggunakan metode *Rectified Linear*

Unit dengan *Stochastic Gradient Descent* pada 2221 Dataset *Colorectal Tumor* menghasilkan perbandingan sebagai berikut:

1. *ReLu* + *SGD* Akurasi = **99.96%.**
2. *ReLu* + *Adam* Akurasi = 99.75%.
3. *ReLu* + *Limited-BFGS-B* Akurasi = 99.83%.
4. *TanH* + *Adam* Akurasi = 99.86%.
5. *TanH* + *SGD* Akurasi = 99.86%.
6. *TanH* + *Limited-BFGS-B* Akurasi = 99.84%.
7. *Sigmoid* + *Adam* Akurasi = 99.86%.
8. *Sigmoid* + *SGD* Akurasi = 99.74%.
9. *Sigmoid* + *Limited-BFGS-B* Akurasi = 99.79%.
10. *Linear* + *Adam* Akurasi = 99.76%.
11. *Linear* + *SGD* Akurasi = 99.85%.
12. *Linear* + *Limited-BFGS-B* Akurasi = 99.55%.

Penelitian ini menunjukkan bahwa *ReLu* + *SGD* menghasilkan akurasi tertinggi.

Pada penelitian yang berjudul "*Accelerated Singular Value Decomposition (ASVD) using momentum based Gradient Descent Optimization*" yang dilakukan oleh (Raghuwanshi dan Pateriya 2021) menggunakan metode *Accelerated Singular Value Decomposition* berdasarkan Optimasi *Gradient Descent* pada Dataset: *MovieLens 100K : 943 User, 1682 Items, FilmTrust : 1508 User, 2071 Items, YahooMovie : 7642 User, 11926 Items* menghasilkan perbandingan nilai *MAE* dan *RMSE* pada *RSVD (Singular Value Decomposition with Using Vanilla Gradient*

Descent), SSVD (Singular Value Decomposition with Using SGD), dan *ASVD (Singular Value Decomposition with Proposed Momentum SGD)* sebagai berikut:

1. MovieLens 100K:

a. MAE = SSVD = ***2.321.***

b. RMSE = SSVD = ***2.742.***

2. FilmTrust:

a. MAE = SSVD = ***2.990.***

b. RMSE = ASVD = ***3.154.***

3. YahooMovie:

a. MAE = ASVD = ***3.365.***

b. *RMSE = ASVD =* ***3.837.***

Penelitian ini menunjukkan bahwa *SSVD* dan *ASVD* menghasilkan nilai *MAE* dan *RMSE* terendah.

Pada penelitian yang berjudul "*Design of Momentum Fractional Stochastic Gradient Descent for Recommender Systems*" yang dilakukan oleh (Khan dkk. 2019) menggunakan metode *Momentum Fractional Stochastic Gradient Descent (mf-SGD)* pada *ML-100K (943 x 1682) Dataset* dan *ML-1M (6040 x 3952) Dataset* menghasilkan perbandingan *RMSE* sebagai berikut :

1. *ML-100K:*

a. *ConvMF* = 1.000.

b. *DBPMF* = 0.990.

c. *DCBPMF* = 0.985.

d. *mF-SGD* = **0.962.**

2. *ML-1M:*
    a. *ConvMF* = 0.980.
    b. *DBPMF* = 0.945.
    c. *DCBPMF* = 0.943.
    d. *mF-SGD* = **0.887.**

Penelitian ini menunjukkan bahwa *mF-SGD* menghasilkan nilai *RMSE* terendah.

Pada penelitian yang berjudul "*Belief-Rule-Base Inference Method Based on Gradient Descent With Momentum*" yang dilakukan oleh (Guan dkk. 2021) menggunakan metode *Stochastic Gradient Descent with Momentum (SGDM)* pada Data *Iris = 150, Wine = 178, Diabetes = 768, Ecoli = 336, Glass = 214, Seeds = 210, Yeat = 1484* menghasilkan perbandingan *average rank* akurasi sebagai berikut:

1. *SGDM-BRB* = **1.71.**
2. *KNN* = 3.28.
3. *BA-EBRB* = 3.66.
4. *MVP-EBRB* = 4.16.
5. *NB* = 4.33.
6. *C4.5* = 5.33.
7. *SRA-EBRB* = 5.42.
8. *SVM* = 6.44.

Penelitian ini menunjukkan bahwa *SGDM-BRB* menghasilkan peringkat *average rank* pertama pada akurasi.

Pada penelitian yang berjudul "*Application of Stochastic Gradient Descent Technique for Method of Moments*" yang dilakukan oleh (Guo dkk. 2020) menggunakan metode Perbandingan *Stochastic Gradient Descent (SGD)* dan *Generalized Minimal Residual Algorithm (GMRES)* pada *Method of Moments (MoM)* pada *PEC Almond* dengan ukuran 1.26m x 0.49 x 0.16m menghasilkan perbandingan waktu konvergensi sebagai berikut:

1. *SGD-MoM* = **9.74 detik.**
2. *GMRES-MoM* = 58.02 detik.

Penelitian ini menunjukkan bahwa *SGD-MoM* menghasilkan waktu konvergensi lebih sedikit.

Pada penelitian yang berjudul "*The Implementation of Gradient Descent Based Methods Using Parallel Computing in R for Regression Tasks*" yang dilakukan oleh (Riza, Ashari, dan Megasari 2018) menggunakan metode *Mini Batch Gradient Descent* pada 1.425 Data sensus *California Housing* menghasilkan perbandingan waktu konvergensi sebagai berikut:

1. *MBGD* : 0.01 detik = 0
2. *SGD* : 0.01 detik = **2**
3. *SAGD* : 0.01 detik = 1
4. *AGD* : 0.01 detik = 1

Penelitian ini menunjukkan bahwa *SGD* menghasilkan waktu konvergensi 0.01 detik paling banyak.

Pada penelitian yang berjudul *"Convolutional Neural Network Combined With Stochastic Parallel Gradient Descent to Decompose Fiber Modes Based on*

Far-Field Measurements" yang dilakukan oleh (Kim, Na, dan Jeong 2023) menggunakan metode *Convolutional Neural Network Combined With Stochastic Parallel Gradient Descent* pada *Fiber Modes* menghasilkan perbandingan nilai *error* sebagai berikut:

1. *CNN*: 0.0444, 0.0472, 0.0249, 0.0381, 0.0499, 0.0352, 0.0189, 0.0320, 0.0494, 0.0404.
2. *CNN/SPGD (CNN HYBRID):* **0.0109, 0.0131, 0.0085, 0.0102, 0.0174, 0.0066, 0.0033, 0.0058, 0.0087, 0.0027.**

Penelitian ini menunjukkan bahwa *CNN/SPGD (CNN HYBRID)* menghasilkan nilai *error* lebih sedikit.

Adapun rekapitulasi penelitian terkait dapat dilihat pada **Tabel II.1** sebagai berikut :

Tabel II.1 Penelitian Terkait

No.	Judul dan Peneliti	Metode	Data	Hasil
1	"*Modified Backpropagation with Added White Gaussian Noise in Weighted Sum for Convergence Improvement*" (Sapkal dan Kulkarni 2018)	*Noise Injection* dengan tambahan *White Gaussian Noise*	*8 Signal to noise* pada 2 bit *iris dataset (iris setosa, iris versicolour, iris virginica)*	1. *BP* = 6050. 2. *NNBP* = **602.5.** Penelitian ini menunjukkan bahwa *Backpropagation* dengan tambahan *Noise Net* manghasilkan waktu konvergensi lebih sedikit.
2	"*AG-SGD : Angle-Based Stochastic Gradient Descent*" (Song, Pons, dan Yen 2021)	*AG-SGD*	Data Sudut 0 – 180 derajat	1. *Vanilla* = 20.00. 2. *Momentum* = 19.92. 3. *RMSProp* = 17.48. 4. *Adam* = 18.93. 5. *Nadam* = 18.40.

No.	Judul dan Peneliti	Metode	Data	Hasil
				6. *AdaMax* = 23.96. 7. *AdaDelta* = 14.62. 8. *AdaGrad* = 24.34. 9. *AMSGrad* = 17.73. 10. *AG-SGD* = **13.71.** Penelitian ini menunjukkan bahwa *AG-SGD* menghasilkan nilai rata – rata *error* terendah.
3	"*Plate Recognition Using Backpropagation Neural Network and Genetic Algorithm*" (Tarigan dkk. 2017)	*Genetic Algorithm*	220 Foto Data Plat Nomor Kendaraan	*1. Genetic Algorithm Backpropagation Neural Network (GABPNN):* a. *Epoch* = **2230.** b. *Training Time* = **3 Menit 9 Detik.** *2. Backpropagation Neural Network (BPNN):* a. Epoch = 3530. b. *Training Time* = 3 Menit 43 Detik. Penelitian ini menunjukkan bahwa *GABPNN* menghasilkan *epoch* dan *training time* lebih sedikit.
4	"*Identifying Colorectal Tumor For Single Cell*	*Rectified Linear Unit*	2221 Dataset *Colorectal Tumor*	1. *ReLu + SGD* Akurasi = **99.96%.**

No.	Judul dan Peneliti	Metode	Data	Hasil
	RNA Sequence Using Rectified Linear Unit With Stochastic Gradient Descent" (Rajesh dkk. 2023)	dengan *Stochastic Gradient Descent*		2. *ReLu + Adam* Akurasi = 99.75%. 3. *ReLu + Limited-BFGS-B* Akurasi = 99.83%. 4. *TanH + Adam* Akurasi = 99.86%. 5. *TanH + SGD* Akurasi = 99.86%. 6. *TanH + Limited-BFGS-B* Akurasi = 99.84%. 7. *Sigmoid + Adam* Akurasi = 99.86%. 8. *Sigmoid + SGD* Akurasi = 99.74%. 9. *Sigmoid + Limited-BFGS-B* Akurasi = 99.79%. 10. *Linear + Adam* Akurasi = 99.76%. 11. *Linear + SGD* Akurasi = 99.85%. 12. *Linear + Limited-BFGS-B* Akurasi = 99.55%. Penelitian ini menunjukkan bahwa *ReLu + SGD* menghasilkan akurasi tertinggi.
5	"*Accelerated Singular Value Decomposition (ASVD) using*	*Accelerated Singular Value Decompos*	Dataset: *MovieLens 100K : 943 User, 1682 Items, FilmTrust*	*1. MovieLens 100K:* *a. MAE = SSVD = **2.321.***

No.	Judul dan Peneliti	Metode	Data	Hasil
	momentum based Gradient Descent Optimization" (Raghuwanshi dan Pateriya 2021)	*ition* berdasarkan Optimasi *Gradient Descent*	*: 1508 User, 2071 Items, YahooMovie : 7642 User, 11926 Items*	b. *RMSE = SSVD = **2.742.*** 2. *FilmTrust:* a. *MAE = SSVD = **2.990.*** b. *RMSE = ASVD = **3.154.*** 3. *YahooMovie:* a. *MAE = ASVD = **3.365.*** b. *RMSE = ASVD = **3.837.*** Penelitian ini menunjukkan bahwa *SSVD* dan *ASVD* menghasilkan nilai *MAE* dan *RMSE* terendah.
6	"*Design of Momentum Fractional Stochastic Gradient Descent for Recommender Systems*" (Khan dkk. 2019)	*Momentum Fractional Stochastic Gradient Descent (mf-SGD)*	*ML-100K (943 x 1682) Dataset* dan *ML-1M (6040 x 3952) Dataset*	1. *ML-100K:* a. *ConvMF* = 1.000. b. *DBPMF* = 0.990. c. *DCBPMF* = 0.985. d. *mF-SGD* = **0.962.** 2. *ML-1M:* a. *ConvMF* = 0.980. b. *DBPMF* = 0.945. c. *DCBPMF* = 0.943. d. *mF-SGD* = **0.887.** Penelitian ini menunjukkan bahwa *mF-SGD*

No.	Judul dan Peneliti	Metode	Data	Hasil
				menghasilkan nilai *RMSE* terendah.
7	"*Belief-Rule-Base Inference Method Based on Gradient Descent With Momentum*" (Guan dkk. 2021)	*Stochastic Gradient Descent with Momentum (SGDM)*	Data *Iris = 150, Wine = 178, Diabetes = 768, Ecoli = 336, Glass = 214, Seeds = 210, Yeat = 1484*	1. *SGDM-BRB* = **1.71.** 2. *KNN* = 3.28. 3. *BA-EBRB* = 3.66. 4. *MVP-EBRB* = 4.16. 5. *NB* = 4.33. 6. *C4.5* = 5.33. 7. *SRA-EBRB* = 5.42. 8. *SVM* = 6.44. Penelitian ini menunjukkan bahwa *SGDM-BRB* menghasilkan peringkat *average rank* pertama pada akurasi.
8	"*Application of Stochastic Gradient Descent Technique for Method of Moments*" oleh (Guo dkk. 2020)	*Stochastic Gradient Descent (SGD)* dan *Generalized Minimal Residual Algorithm (GMRES)*	*Method of Moments (MoM)* pada *PEC Almond* dengan ukuran 1.26m x 0.49 x 0.16m	1. *SGD-MoM* = **9.74 detik.** 2. *GMRES-MoM* = 58.02 detik. Penelitian ini menunjukkan bahwa *SGD-MoM* menghasilkan waktu konvergensi lebih sedikit.
9	"*The Implementation of Gradient Descent Based Methods Using Parallel Computing in R for Regression Tasks*" (Riza, Ashari, dan Megasari 2018)	*Mini Batch Gradient Descent*	1.425 Data sensus *California Housing*	1. *MBGD* : 0.01 detik = 0 2. *SGD* : 0.01 detik = **2** 3. *SAGD* : 0.01 detik = 1 4. *AGD* : 0.01 detik = 1 Penelitian ini menunjukkan bahwa *SGD* menghasilkan

No.	Judul dan Peneliti	Metode	Data	Hasil
				waktu konvergensi 0.01 detik paling banyak.
10	*"Convolutional Neural Network Combined With Stochastic Parallel Gradient Descent to Decompose Fiber Modes Based on Far-Field Measurements"* (Kim, Na, dan Jeong 2023)	*Convolutional Neural Network Combined With Stochastic Parallel Gradient Descent*	*Fiber Modes*	1. *CNN*: 0.0444, 0.0472, 0.0249, 0.0381, 0.0499, 0.0352, 0.0189, 0.0320, 0.0494, 0.0404. 2. *CNN/SPGD (CNN HYBRID)*: **0.0109, 0.0131, 0.0085, 0.0102, 0.0174, 0.0066, 0.0033, 0.0058, 0.0087, 0.0027.** Penelitian ini menunjukkan bahwa *CNN/SPGD (CNN HYBRID)* menghasilkan nilai *error* lebih sedikit.

II.2. Normalisasi Data

Dalam teknik ini, atribut akan diubah skalanya dari domain aslinya ke rentang nilai baru, seperti antara (0,1) yang dapat dilihat pada persamaan 2.1 sebagai berikut (Al Shalabi dan Shaaban 2006; Eesa dan Arabo 2017):

$$nv = f(v) = \frac{v - min\,(v)}{max\,(v) - min\,(v)} \quad (2.1)$$

Keterangan pada persamaan 2.1 dapat dilihat sebagai berikut:

nv : Nilai hasil normalisasi.

$f(v)$: Fungsi normalisasi.

v : Nilai yang akan dinormalisasi.

$max\ (v)$: Nilai Tertinggi pada himpunan data v.

$min\ (v)$: Nilai Terendah pada himpunan data v.

II.3. Backpropagation

Jaringan *Backpropagation* adalah jaringan yang dilatih dengan memberikan contoh algoritma yang ingin dilakukan, dan menyesuaikan bobot jaringan sehingga setelah dilatih, akan menghasilkan *output* yang diinginkan untuk sebuah inputan *backpropagation* (Majeed Sadeq, Aziz Qadir, dan Hassan Abbas 2023).

Keunggulan dari *Backpropagation* adalah dapat memberikan hasil yang akurat dengan data yang tidak terlalu besar dimana sebagian besar arsitektur dari jaringan saraf tiruan memerlukan *input* data dalam jumlah yang besar (Chen dkk. 2021).

Backpropagation dapat memberikan hasil prediksi dengan akurasi yang baik sehingga merupakan algoritma yang umum digunakan di dalam proses klasifikasi (Nápoles, Jastrzębska, dan Salgueiro 2021).

Pelatihan pada *neural network* membutuhkan 3 operasi yang berurutan: *forward propagation, backpropagation,* dan memperbarui bobot. Dalam *forward propagation,* data dipropagasikan dari *input layer* menuju *output layer*. Setiap *neuron* akan menjumlahkan total bobot dari *input neuron* yang terhubung dari *layer* sebelumnya, dan kemudian menambahkan dengan sebuah *bias* yang dapat dilihat pada persamaan 2.2 sebagai berikut (Park dan Suh 2022):

$$u_j^l = \sum_{i=1}^{N} w_{ij}^l x_{ij}^l + Bias^1 \quad (2.2)$$

Keterangan pada persamaan 2.2 dapat dilihat sebagai berikut:

u_j^l : *Output* perhitungan *feedforward*.

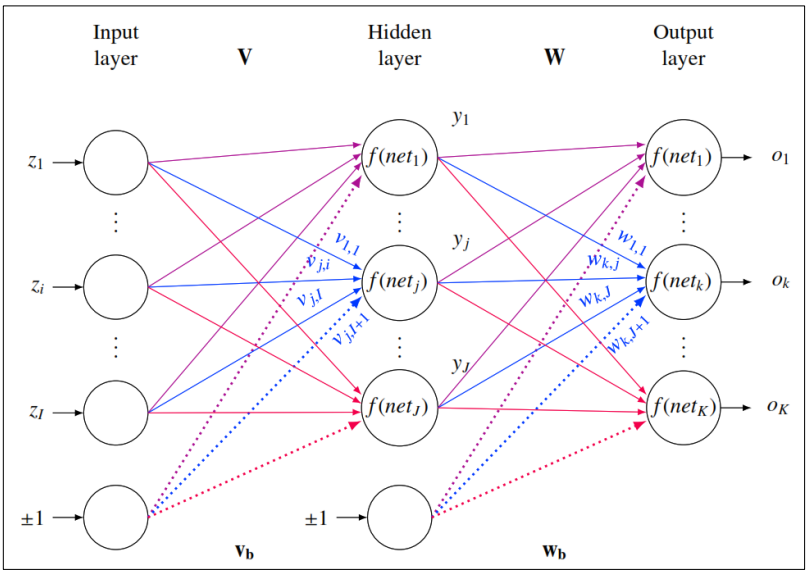
$\sum_{i=1}^{N}$: Penjumlahan berulang sebanyak nilai N.

w_{ij}^l : Bobot pada nilai x ke layer selanjutnya.

x_{ij}^l : Nilai inputan ke *layer* selanjutnya.

$Bias^1$: Nilai Bias.

Backpropagation sangat populer pada algoritma *supervised learning*, arsitektur dari *multilayer perceptron* dapat dilihat pada **Gambar II.1** sebagai berikut (Sapkal dan Kulkarni 2018):



Gambar II.1 Arsitektur *Multilayer Perceptron*

(Sumber : Sapkal dan Kulkarni 2018)

Gambar II.1 menunjukkan arsitektur *multilayer perceptron* yang terdiri dari *input layer, hidden layer,* dan *output layer.*

Backpropagation digunakan untuk menyesuaikan bobot (w_{ij}^{l}) dengan menghitung turunannya. Turunan pada *output layer* dapat dilihat pada persamaan 2.3 sebagai berikut (Park dan Suh 2022):

$$\frac{dE}{dy_z^o} = y_z^o - t_z \tag{2.3}$$

Keterangan pada persamaan 2.3 dapat dilihat sebagai berikut:

$\frac{dE}{dy_z^o}$: Turunan *error* terhadap turunan *output error*

dE : Turunan nilai *error*.

dy_z^o : Turunan nilai *output*.

y_z^o : Hasil prediksi *feedforward*.

t_z : Nilai sebenarnya.

Hasil perhitungan berbeda antara hasil *forward propagation* (y_z^o) dengan target *output* (t_z). Turunan dari *error* dengan perhitungan bobot dapat dilihat pada persamaan 2.4 sebagai berikut (Park dan Suh 2022):

$$\frac{dE}{dy_k^l} = \sum_z w_{kz}^{l+1} \frac{dE}{dy_z^{l+1}} \tag{2.4}$$

Keterangan pada persamaan 2.4 dapat dilihat sebagai berikut:

$\frac{dE}{dy_z^o}$: Turunan dari fungsi *error* terhadap turunan output.

$\sum_z w_{kz}^{l+1} \frac{dE}{dy_z^{l+1}}$: Perulangan setiap turunan dari fungsi *error* terhadap turunan *output*.

II.4. Fungsi Aktivasi

Fungsi aktivasi membantu dalam pelatihan agar menjadi lebih baik, proses pelatihan dan kemampuan generalisasi menjadi lebih baik. Fungsi akivasi mengendalikan setiap unit didalam *layer* dengan bekerja pada bobot dan *bias*. Proses ini menggabungkan *non linear* dalam *output* dari setiap *neuron*. Bobot diupdate berdasarkan *error output.* Proses dari *backpropagation* mendukung fungsi aktivasi dengan memberikan *gradient* (Singh dkk. 2023).

Output dari pelatihan akan bergerak melalui fungsi aktivasi yang ditentukan untuk melewati *layer* selanjutnya. Kebanyakan fungsi aktivasi yang digunakan adalah *Rectified Linear Unit (ReLU), Tanh,* dan *Sigmoid* (Park dan Suh 2022)

Fungsi aktivasi *sigmoid* terutama digunakan untuk klasifikasi multi-klasifikasi karena memetakan nilai *input* dalam rentang (0,1) yang pada dasarnya probabilitasnya milik *class* yang diberikan yang dapat dilihat pada persamaan 2.5 sebagai berikut (Elfwing, Uchibe, dan Doya 2018; Singh dkk. 2023):

$$\sigma(x) = \left(\frac{1}{1+\exp(-x)}\right) \quad (2.5)$$

Keterangan pada persamaan 2.5 dapat dilihat sebagai berikut:

$\sigma(x)$: *Output* yang akan diberikan fungsi aktivasi.

$\exp(-x)$: Bilangan *exponen* yang bernilai 2.71.

II.5. Error

Ketika nilai output jaringan tidak sama dengan nilai sebenarnya ada nilai loss yang dapat dilihat pada persamaan 2.6 sebagai berikut (Chen dkk. 2021) :

$$Loss = \frac{1}{2}\sum_{i=1}^{m}(yi^{predict} - yi^{data})^2 \quad (2.6)$$

Keterangan pada persamaan 2.6 dapat dilihat sebagai berikut:

$Loss$: Nilai *error*.

$yi^{predict}$: Nilai prediksi yang dihasilkan model.

yi^{data} : Nilai sebenarnya atau target.

II.6. Stochastic Gradient Descent

Pada bidang *neural network, stochastic gradient descent* sering digunakan sebagai metode yang efektif untuk mempercepat hasil konvergensi. Menghasilkan *gradient* baru dari gradient sebelumnya dimana metode ini diadopsi dari banyak algoritma optimisasi yang ada (Song, Pons, dan Yen 2021).

Algoritma pembelajaran pada *Backpropagation* bergantung pada metode *Gradient Descent* untuk mengoptimalkan proses pelatihan. Fungsi jaringan yang interaktif mengurangi *error* berdasarkan perubahan bobot jaringan yang konstan (Q. Zhao dkk. 2021).

Stochastic Gradient Descent : x_t menjadi sebuah contoh acak dari *data set* pada sebuah iterasi *t,* $J_\theta(x_t)$ adalah fungsi objektif yang mengevaluasi pada data x_t dengan parameter θ, $g_t = \nabla_\theta J_\theta(x_t)$ adalah *gradient*, dan α adalah *learning rate* (Ilboudo, Kobayashi, dan Sugimoto 2022). Algoritma *SGD* mengupdate θ_{t-1} to θ_t melalui persamaan 2.7 sebagai berikut (Robbins dan Monro 1951; Ilboudo, Kobayashi, dan Sugimoto 2022) :

$$\theta_t = \theta_{t-1} - \alpha g_t \tag{2.7}$$

Keterangan pada persamaan 2.7 dapat dilihat sebagai berikut:

θ_t : Bobot / Bias baru.

θ_{t-1} : Bobot / Bias lama.

α : *Learning Rate*.

g_t : *Gradient* setiap inputan.

II.7. Mean Square Error (MSE)

Loss Function pada model dapat dinyatakan dengan *Mean Square Error (MSE)* pada persamaan 2.8 sebagai berikut (Qu dkk. 2023) :

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(yi^{predict} - yi^{data})^2 \quad (2.8)$$

Keterangan pada persamaan 2.8 dapat dilihat sebagai berikut:

MSE : Rentang nilai prediksi dengan nilai sebenarnya.

$\frac{1}{N}\sum_{i=1}^{N}$: Penjumlahan dari semua nilai dalam *dataset*.

$yi^{predict}$: Nilai prediksi yang dihasilkan model.

yi^{data} : Nilai sebenarnya atau target.

II.8. Pengujian

Untuk mengetahui performa dari model yang telah dilatih akan dilakukan perhitungan *accuracy, precision, recall,* dan *f1* yang dilihat sebagai berikut :

1. Accuracy

   Ini adalah matriks yang digunakan untuk menjelaskan bagaimana model bekerja di setiap classnya yang dapat dilihat pada persamaan 2.9 sebagai berikut (Rajesh dkk. 2023) :

$$Accuracy = \frac{True_{Positive} + True_{Negative}}{True_{Positive} + True_{Negative} + False_{Positive} + False_{Negative}} \quad (2.9)$$

Keterangan pada persamaan 2.9 dapat dilihat sebagai berikut:

$Accuracy$: Mengukur sejauh mana model memprediksi dengan benar.

$True_{Positive}$: Jumlah sampel yang benar diprediksi sebagai *positive*.

$True_{Negative}$: Jumlah sampel yang benar diprediksi sebagai *negative*.

$False_{Positive}$: Jumlah sampel yang salah diprediksi sebagai *positive*.

$False_{Negative}$: Jumlah sampel yang salah diprediksi sebagai *negative*.

2. Precision

Ini adalah rasio Benar Positif dan semua prediksi benar yang dapat dilihat pada persamaan 2.10 sebagai berikut (Rajesh dkk. 2023) :

$$Precision = \frac{True_{Positive}}{True_{Positive} + False_{Positive}} \quad (2.10)$$

Keterangan pada persamaan 2.10 dapat dilihat sebagai berikut:

$Precision$: Rasio benar positif dan semua prediksi benar.

$True_{Positive}$: Jumlah sampel yang benar diprediksi sebagai *positive*.

$False_{Positive}$: Jumlah sampel yang salah diprediksi sebagai *positive*.

3. Recall

Ini adalah proporsi positif yang diidentifikasi dengan benar yang semakin banyak sampel positif terdeteksi maka nilai *recall* semakin tinggi yang dapat dilihat pada persamaan 2.11 sebagai berikut (Rajesh dkk. 2023) :

$$Recall = \frac{True_{Positive}}{True_{Positive} + False_{Negative}} \quad (2.11)$$

Keterangan pada persamaan 2.11 dapat dilihat sebagai berikut:

$Recall$: Proposi *positive* yang diidentifikasi dengan benar.

$True_{Positive}$: Jumlah sampel yang benar diprediksi sebagai *positive*.

$False_{Negative}$: Jumlah sampel yang salah diprediksi sebagai *negative*.

4. F1

Ini adalah rasio seimbang diantara *precision* dan *recall* yang dapat dilihat pada persamaan 2.12 sebagai berikut (Rajesh dkk. 2023) :

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (2.12)$$

Keterangan pada persamaan 2.12 dapat dilihat sebagai berikut:

$F1$: Rasio seimbang diantara *precision* dan *recall*

$Precision$: Nilai *precision.*

$Recall$: Nilai *recall.*

II.9. MATLAB

Matlab (Matrix Laboratory) adalah program yang dapat digunakan untuk kepentingan edukasi dan pembelajaran penelitian. Dengan program ini, simulasi waktu, sistem *linear*, analisis modal, analisis faktor dan visualisasi, optimasi penemapatan *controller*, seleksi umpan balik, analisis respon frekuensi, dan desain *control* dapat dilakukan (Abdulrahman 2020).

Matlab adalah sebuah bahasa pemrograman yang sangat digunakan pada domain ilmiah dan teknik oleh para *enginer*, ilmuwan, dan peneliti (Reis, Gralha, dan Monteiro 2022).