Menentukan Solusi Persamaan Diferensial Parsial dengan Algoritma *Backpropagation Neural Network*

SKRIPSI

HERDIANTO

H011191067

PROGRAM STUDI MATEMATIKA DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

2023

Menentukan Solusi Persamaan Diferensial Parsial dengan Algoritma *Backpropagation Neural Network*

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Sains pada Program Studi Matematika Departemen Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

HERDIANTO

H011191067

PROGRAM STUDI MATEMATIKA DEPARTEMEN MATEMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS HASANUDDIN

MAKASSAR

OKTOBER 2023

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama	: HERDIANTO
Nim	: H011191067
Program Studi	: Matematika
Jenjang	: S1

Menyatakan dengan ini bahwa karya tulisan saya yang berjudul:

Menentukan Solusi Persamaan Diferensial Parsial dengan Algoritma *Backpropagation Neural Network*

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilan alihan tulisan orang lain bahwa tulisan skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Makassar, 14 November 2023

Yang menyatakan,

HERDIANTO
NIM. H011191067

LEMBARAN PENGESAHAN

Menentukan Solusi Persamaan Diferensial Parsial dengan Algoritma *Backpropagation Neural Network*

Disusun dan diajukan oleh

HERDIANTO

H011191067

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Hasanuddin

Pada tanggal, 14 November 2023

Dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama	Pembimbing Pertama
Naimah Aris, S.Si., M.Math	**A. Muh Amil Siddik, S.Si, M.Si.**
NIP. 19711003 199702 2 001	**NIP. 19911003 201903 1 015**

Ketua Program Studi,

Prof. Dr. Nurdin, S.Si., M.Si.
NIP. 19700807 200003 1 002

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN UNIVERSITAS HASANUDDIN DEPARTEMEN MATEMATIKA FAKULTAS MIPA

KATA PENGANTAR

Puji Tuhan, penulis panjatkan puji syukur ke hadirat Tuhan yang Maha Esa atas segala limpahan rahmat, nikmat, dan karunia-Nya sehingga tugas akhir ini dapat terselesaikan dengan lancar yang berjudul **"Menentukan Solusi Persamaan Diferensial Parsial dengan Algoritma *Backpropagation Neural Network*"**. Dengan berbagai rintangan yang dihadapi saat menyelesaikan tugas ini, tidak lupa untuk penulis mengucapkan terima kasih atas kontribusi dan bantuannya kepada :

1. Bapak **Prof. Dr. Ir. Jamaluddin Jompa, M.Sc.** selaku Rektor Universitas Hasanuddin berserta seluruh jajarannya, serta **Bapak Dr. Eng. Amiruddin** selaku Dekan Fakultas Matematika Dan Ilmu Pengetahuan Alam beserta jajarannya.
2. Bapak **Prof. Dr. Nurdin, S.Si., M.Si.** selaku Ketua Departemen Matematika Fakultas Matematika Dan Ilmu Pengetahuan Alam Universitas Hasanuddin beserta Bapak dan Ibu **Dosen Departemen Matematika** yang telah membantu dan memudahkan penulis dalam berbagai hal administrasi.
3. Ibu **Naimah Aris, S.Si., M.Math** dan Bapak **A. Muh Amil Siddik, S.Si, M.Si.** selaku dosen pembimbing yang dengan sabar telah memberikan saran, kritik, dan masukan yang bermanfaat sehingga skripsi ini dapat tersusun dengan baik.
4. Bapak **Dr. Agustinus Ribal, S.Si.,M.Sc** dan bapak **Edy Saputra Rusdi, S.Si.,M.Si.** selaku dosen penguji I dan dosen penguji II yang telah memberikan saran-saran yang membangun demi kesempurnaan skripsi yang telah disusun untuk menjadi lebih baik lagi.
5. Kedua orang tua penulis, Bapak **Hermanto** dan Ibu **Herniati** yang selalu memberikan doa dan dukungan selama ini dan juga terkhusus untuk bapak yang memberikan masukan seputar bagaimana dalam menyusun skripsi dengan baik.
6. Adik penulis, **vikti kristiani**, dan keluarga besar yang telah memberikan dukungan dan doa sampai titik terselesaikannya penyusunan skripsi.
7. Tante **Rice'** dan om **Yos** yang terus mendukung penulis dalam menyelesaikan skripsi ini dengan baik.
8. Teman-teman seperjuangan, Samuel, ayub, Heri, Fatha, Jeki, ferdi, ade, Rifqy, kiply, Rozzaq, Ichsan, Resa, Wawan, Toni, Ilham, Hanif, Rais, Rabil, Zidan, Fian, dan

seluruh teman **Prodi Matematika 2019** yang telah membantu dan memberi semangat dalam menyusun skripsi ini dan berjuang bersama-sama hingga saat ini.

9. Teman-teman **KKNT Perhutanan Sosial Toraja Utara,** terkhusus **Bokin Pride,** Ady, Mas Rio, Dody, Angguna, Irene, Ririn dan Mima yang terus mendukung penulis sampai bisa menyelesaikan skripsi ini dengan baik.
10. Semua pihak yang terlibat dalam penulisan skripsi ini yang belum bisa disebutkan satu per satu yang telah membantu dalam doa serta mendukung dalam proses penulisan skripsi ini.

Semoga Tuhan Yesus memberkati dan memberikan rahmat dan berkat-Nya kepada semua pihak yang telah membantu menyelesaikan skripsi ini dan semoga kebaikan akan dibalaskan dengan yang lebih baik lagi. Walaupun masih jauh dari kata sempurna, tetapi dengan tulisan ini semoga dapat bermanfaat bagi pembacanya.

Makassar, Oktober 2023

HERDIANTO

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMISI

Sebagai sivitas akademik Universitas Hasanuddin, saya yang bertanda tangan di bawah ini:

Nama	: HERDIANTO
Nim	: H011191067
Program Studi	: Matematika
Departemen	: Matematika
Fakultas	: Matematika dan Ilmu Pengetahuan Alam
Jenis Karya	: Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Hasanuddin **Hak Bebas Royalti Noneksklusif *(Non-exclusive Royalty- Free Right)* atas** karya ilmiah saya yang berjudul:

Menentukan Solusi Persamaan Diferensial Parsial dengan Algoritma *Backpropagation Neural Network*

Beserta perangkat yang ada (jika diperlukan). Terkait dengan hal di atas, maka pihak Universitas berhak menyimpan, mengalih-media/format-kan, mengelolah dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak cipta.

Demikian pernyataan ini saya buat dengan sebenarnya,
Dibuat di Makassar pada tanggal, Oktober 2023

Yang menyatakan,

HERDIANTO

ABSTRAK

Persamaan diferensial digunakan untuk menggambarkan model dari berbagai masalah dan fenomena dalam kehidupan sehari-hari. Menyelesaikan persamaan diferensial parsial dapat dilakukan dengan memperoleh solusi analitiknya. Penelitian ini membahas penyelesaian masalah matematika yaitu Persamaan Gelombang, Persamaan Burger *inviscid* dan Persamaan Panas yang masing-masing sudah memiliki solusi analitik. Metode yang digunakan untuk memperkirakan solusi dari persamaan diferensial parsial adalah *artificial neural network*, dan salah satu yang paling populer adalah algoritma backpropagation neural network. Metode ini bekerja berdasarkan bobot-bobot masukan yang diberikan, berdasarkan bobot-bobot tersebut model ini mampu belajar mencapai solusi analitik dari suatu PDP dengan tingkat kesalahan tertentu (RMSE). Dengan menggunakan arsitektur *multilayer net* serta menggunakan fungsi aktivasi *sigmoid*. Hasil yang diperoleh bahwa semakin banyak neuron pada hidden layer maka akan semakin cepat solusi konvergen ke solusi analitiknya namun RMSE cukup besar dan sebaliknya semakin sedikit neuron pada hidden layer makan semakin lama solusi konvergen ke solusi analitik PDP tetapi nilai RMSE dari model cukup kecil.

Kata Kunci: persamaan diferensial parsial, *backpropagation neural network* , Persamaan Panas 1-D, Persamaan Gelombang 1-D, Persamaan Burger inviscid, RMSE

ABSTRACT

Differential equations are used to describe models of various problems and phenomena in everyday life. Solving partial differential equations can be done by obtaining analytical solutions. This research discusses solving mathematical problems, namely the wave equation, the inviscid Burger equation and the heat equation, each of which has an analytical solution. The method used to estimate solutions to partial differential equations is artificial neural networks, and one of the most popular is the backpropagation neural network algorithm. This method works based on the given input weights, based on these weights this model is able to learn to reach an analytical solution of a PDP with a certain error rate (RMSE). By using a multilayer net architecture and using a sigmoid activation function. The results obtained show that the more neurons in the hidden layer, the faster the solution will converge to the analytical solution, but the RMSE is quite large and conversely, the fewer neurons in the hidden layer, the longer it will take for the solution to converge to the PDP analytical solution, but the RMSE value of the model is quite small.

Keywords: Partial differential equations, *backpropagation neural network*, one-dimensional heat equation, one-dimensional wave equation, inviscid Burger equation, RMSE.

Daftar isi

JUDUL i

PERNYATAAN KEASLIAN iii

LEMBAR PENGESAHAN iv

KATA PENGANTAR v

PERNYATAAN PERSETUJUAN PUBLIKASI vii

ABSTRAK viii

ABSTRACT ix

DAFTAR ISI x

DAFTAR SINGKATAN DAN SIMBOL xi

BAB I PENDAHULUAN 1

1.1. Latar Belakang 1
1.2. Perumusan Masalah 2
1.3. Batasan Masalah 3
1.4. Tujuan Penelitian 3
1.5. Manfaat Penelitian 3
1.6. Sistematika Penulisan 3

BAB II TINJAUAN PUSTAKA 5

2.1 Persamaan Diferensial Parsial 5
2.2 Solusi Persamaan Diferensial Parsial 5
2.3 Klasifikasi Persamaan diferensial Parsial 7
2.4 Persoalan Nilai Awal dan nilai Batas 10
2.5 Metode *Artificial Neural Network* 10
2.6 Algoritma *Backpropagation* 19
2.7 Contoh soal menentukan solusi PDP dengan Algoritma BPNN 24

BAB III METODOLOGI PENELITIAN 32

3.1. Metode Penelitian 32
3.2. Waktu dan Tempat Penelitian 32
3.3. Prosedur Penelitian 32
3.4. Alur Penelitian 34

BAB IV HASIL DAN PEMBAHASAN 35

4.1. Model *Backpropagation Neural Network* dan *Loss function* 35
4.2. *Training* BPNN dan Solusi PDP dengan Algoritma BPNN 37
4.3. Perbandingan Solusi Algoritma BPNN dengan Solusi analitik 64

BAB V PENUTUP 67

5.1. Kesimpulan 67
5.2. Saran 68

DAFTAR PUSTAKA 69

LAMPIRAN 72

Daftar Singkatan dan Simbol

PDP	: Persamaan diferensial parsial.
BPNN	: *Backpropagation Neural Network.*
ANN	: *Artificial Neural Network.*
MSE	: *Mean Square Error.*
SGD	: *Stochastic gradient descent*
$u(x,t)$	: Solusi analitik PDP.
$\hat{u}(x,t)_{out}$	: Solusi numerik PDP dengan metode BPNN.
Z_i	: *Input* ke-*i.*
$h_{j,k}$	: *Hidden layer* ke-*j*, layer ke-*k.*
$W_{i,j}$	: Bobot *W* ke-*i*, *hidden layer* ke-*j.*
V_i	: Bobot *V* k-*i.*
b_i	: Bias ke-*i.*
α	: *Learning rate*.
ϕ	: Fungsi aktivasi *sigmoid.*
ϕ'	: Turunan fungsi aktivasi *sigmoid*
β	: Skala Nguyen-Windrow $\left(\beta = 0{,}7(p)^{\frac{1}{n}}\right)$, dengan p adalah jumlah unit *hidden layer,* dan *n* adalah jumlah unit *input layer.*
n	: Jumlah *input layer.*

BAB I
PENDAHULUAN

1.1. Latar Belakang

Persamaan diferensial adalah salah satu cabang ilmu matematika yang menarik untuk dibahas, karena terapannya memiliki dampak sangat besar dalam pengembangan matematika *modern.* Persamaan diferensial digunakan untuk menggambarkan model dari berbagai masalah dan fenomena dalam kehidupan sehari-hari. Misalnya dalam kasus ideal, dimana satu-satunya variabel independen adalah waktu, hukum kedua newton secara efektif merupakan persamaan diferensial biasa (PDB) yang dapat dipecahkan untuk menghitung kecepatan objek pada setiap waktu. Akan tetapi dalam situasi yang lebih kompleks, dengan banyak gaya yang bekerja pada bagian-bagian yang bergerak pada sistem yang rumit dari waktu ke waktu, ilmuwan menggunakan persamaan diferensial parsial (PDP) untuk menggambarkan fenomena yang melibatkan banyak variabel bebas. PDP adalah persamaan diferensial yang melibatkan turunan parsial yang mengandung dua atau lebih variabel (Kusuma, 2018).

Menyelesaikan PDP dapat dilakukan dengan cara memperoleh solusi analitiknya, tetapi banyak kasus yang solusi analitiknya belum ditemukan dan hanya diperoleh solusi numeriknya saja. Misalnya, dalam kasus sains dan teknik banyak PDP yang sulit untuk dipecahkan, oleh karena itu dibutuhkan cara yang lebih baik dan lebih efisien untuk menyelesaikan persamaan PDP tersebut (Ananthawamy, 2021).

Para ilmuwan telah mengembangkan metode baru yaitu jaringan saraf tiruan untuk memperkirakan solusi dari persamaan diferensial parsial. Jaringan saraf tiruan atau sering dikenal dengan *neural network* didefinisikan sebagai sistem pemrosesan informasi yang mempunyai karakteristik menyerupai jaringan saraf manusia. Seperti halnya jaringan saraf manusia yang bekerja sesuai dengan jenis rangsangan atau pesan saraf yang diterima, *Neural network* juga bekerja sesuai dengan bobot-bobot masukan yang diberikan. Berdasarkan bobot-bobot tersebut *neural network* mampu memprediksi solusi dari suatu PDP dengan tingkat kesalahan (*loss function*) tertentu.

Penggunaan jaringan saraf tiruan sudah diterapkan di berbagai bidang ilmu pengetahuan, misalnya dalam keilmuan yaitu *Artificial neural network forsolving ordinary and partial differensial equation* (Lagaris, dkk), dalam bidang kesehatan yaitu pengembangan metode *neural network* untuk menentukan karakter seseorang (Pristanti dan Windana, 2015), dan Selanjutnya dalam bidang ekonomi jaringan saraf tiruan algoritma *backpropagation* dalam memprediksi ketersediaan komoditi pangan provinsi Riau (Cynthia dan Ismanto, 2017).

Dalam memprediksi sebuah solusi PDP, tentunya diharapkan nilai dari *loss function* yang minimum. Oleh karena itu, *neural network* membutuhkan metode pembelajaran yang dapat mengetahui apakah hasil dari prediksi tersebut sudah sesuai dengan solusi yang diinginkan atau biasa disebut dengan metode *supervised learning*. Sala satu metode *supervised learning* yang bisa digunakan yaitu algoritma *Backpropagation Neural Network.* Algoritma *Backpropagation Neural Network* (BPNN) merupakan metode penyesuaian bobot-bobot *neural network* dengan arah mundur berdasarkan nilai *loss function* (Windarto, Lubis, dan Solikhun, 2018).

Berdasarkan penelitian yang dilakukan oleh Wijaya, dan kawan-kawan pada tahun 2007 tentang perbandingan penyelesaian persamaan diferensial biasa menggunakan metode *Backpropagation,* metode Euler, metode Heun, dan Runge-Kutta orde 4 yang memperoleh kesimpulan bahwa metode *Backpropagation Neural Network* (BPNN) memiliki tingkat akurasi yang lebih tinggi dibandingkan dengan metode Euler, Heun dan Runge-Kutta orde 4. Berdasarkan penelitian tersebut, peneliti ingin menentukan solusi PDP dengan algoritma *Backpropagation Neural Network* (BPNN) kemudian membandingkan solusi PDP tersebut dengan solusi analitik yang telah diperoleh oleh peneliti sebelumnya, dan dituangkan dalam penelitian yang berjudul **Menentukan Solusi Persamaan Diferensial Parsial dengan Algoritma *Backpropagation Neural Network*.**

1.2. Perumusan Masalah

Rumusan masalah berdasarkan latar belakang di atas antara lain:

1. Bagaimana cara menentukan solusi persamaan diferensial parsial menggunakan algoritma *Backpropagation Neural Network* (BPNN).

2. Bagaimana perbandingan antara solusi *Backpropagation Neural Network* (BPNN) dengan solusi analitik PDP yang berikan.

1.3. Batasan Masalah

Masalah penelitian ini dibatasi pada Persamaan Gelombang 1-D, Persamaan Burger *inviscid,* Persamaan Panas 1-D, dengan masing-masing persamaan memiliki solusi analitik yang diperoleh dari peneliti sebelumnya. Adapun metode yang digunakan adalah algoritma *Backpropagation Neural Network* dengan inisialisasi Nguyen-Windrow. Adapun arsitektur *Artificial Neural Network* (ANN) yang digunakan adalah *multilayer net* dengan dua *hidden layer*.

1.4. Tujuan Penelitian

Tujuan penelitian ini ialah:

1. Menentukan solusi dari persamaan diferensial parsial (PDP) menggunakan algoritma *Backpropagation neural network* (BPNN).
2. Melakukan perbandingan antara *backpropagation neural network* (BPNN) dengan solusi analitik PDP yang diberikan.

1.5. Manfaat Penelitian

Manfaat yang diharapkan dalam penelitian ini dapat menambah pemahaman dan menjadi sumber referensi dalam mengembangkan ilmu matematika di era *modern* ini, khususnya pada matematika terapan.

1.6. Sistematika penulisan

Sistematika penulisan yang digunakan dalam penelitian ini terdiri dari lima bagian, dengan masing-masing dibagi dalam beberapa subbab. Adapun rincian sistematika penulisan pada penelitian ini adalah sebagai berikut.

BAB I PENDAHULUAN

Bab ini mencakup latar belakang, rumusan masalah. batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini mencakup pemaparan secara singkat mengenai konsep dasar yang menunjang pembahasan masalah, yaitu definisi-definisi dan istilah istilah yang berkaitan dengan algoritma *Backpropagation Neural Network* (BPNN).

BAB III METODOLOGI PENELITIAN

Bab ini mencakup metode penelitian, waktu dan tempat penelitian, dan prosedur penelitian.

BAB IV PEMBAHASAN

Pada bab ini disajikan pembahasan dari tugas akhir yakni menentukan solusi persamaan diferensial parsial dengan algoritma *Backpropagation Neural Network* (BPNN) dan bagaimana perbandingan antara solusi analitik dan solusi numerik dengan metode algoritma BPNN.

BAB V PENUTUP

Pada bab ini menyajikan kesimpulan hasil penelitian dan saran yang ditujukan bagi peneliti selanjutnya untuk mengembangkan penelitian yang telah dikaji dalam tugas akhir ini.

BAB II
TINJAUAN PUSTAKA

2.1. Persamaan diferensial parsial

Persamaan diferensial parsial muncul secara alamiah dalam berbagai masalah yang melibatkan fungsi dua variabel atau lebih. Pada awalnya persamaan diferensial parsial (PDP) muncul dalam fenomena fisika di ruang lingkup fluida dinamika, elektrisitas, elastisitas, magnetika, mekanika, optik, dan aliran panas. Pada perkembangannya dewasa ini, persamaan diferensial parsial beserta konsepnya muncul dalam berbagai model disiplin ilmu (Kusuma, 2018). Secara umum, persamaan diferensial parsial dituliskan sebagai berikut:

$$f\left(x, y, \dots, u, u_x, u_y, \dots, u_{xx} u_{yy}, \dots\right) = 0. \qquad (2.1)$$

Persamaan diferensial mempunyai banyak notasi, untuk memudahkan notasi yang digunakan dalam penelitian ini adalah *notasi indeks*. Berikut ini penulisan fungsi dengan *notasi indeks*:

$$u = u(x, y)$$

$u = u(x, y, z)$, dan seterusnya.

dituliskan dengan fungsi *u* saja, secara implisit diartikan sebagai fungsi dengan masing-masing dua variabel, tiga variabel, dan seterusnya. Selanjutnya, penulisan pada turunan parsial *u* dituliskan sebagai berikut:

$\frac{\partial u}{\partial x}$ dituliskan sebagai u_x

$\frac{\partial^2 u}{\partial x^2}$ dituliskan sebagai u_{xx}

$\frac{\partial^2 u}{\partial x \partial y}$ dituliskan sebagai u_{xy}, dan seterusnya.

2.2. Solusi persamaan diferensial parsial (PDP)

Suatu fungsi u dikatakan solusi dari suatu PDP jika fungsi u tersebut memenuhi PDP yang diberikan, maka ruas kiri dan ruas kanan PDP tersebut harus sama.

Contoh 2.7:

Tunjukkan bahwa $u(x, t) = sin\ x\ e^{-4t}$ adalah solusi dari PDP $u_t = 4u_{xx}$

Jawab:

Diketahui,

$$u(x,t) = sin\ x\ e^{-4t},$$

diperoleh,

$$u_t = -4sin\ x\ e^{-4t}$$
$$u_x = cos\ x\ e^{-4t}$$
$$u_{xx} = -sin\ x\ e^{-4t}$$

sehingga,

$$u_t = 4u_{xx}$$
$$-4\ sin\ x\ e^{-4t} = 4(-sin\ x\ e^{-4t}).$$

Jadi, karena ruas kiri dan ruang kanan PDP sama, maka fungsi $u(x,t) = sin\ x\ e^{-4t}$ yang diberikan merupakan solusi dari $u_t = 4u_{xx}$.

Pada penelitian ini, akan dilakukan simulasi pada 3 persamaan diferensial parsial yaitu Persamaan Gelombang 1-D, persamaan Burger *inviscid*, dan Persamaan Panas 1-D.

1. Persamaan Gelombang 1-D

Persamaan Gelombang 1-D yang akan ditentukan solusinya pada penelitian ini dinyatakan sebagai berikut:

$$u_{tt} = c^2 u_{xx} \quad -\infty < x < \infty,\ t > 0 \tag{2.2}$$

dengan syarat awal

$$u(x,0) = \sin(x)$$
$$u_t(x,0) = 0 \tag{2.3}$$

dengan *c* adalah konstanta positif

Berdasarkan penelitian yang dilakukan oleh Noor, dkk memperoleh solusi analitik Persamaan Gelombang 1-D dengan syarat awal (2.3) yaitu sebagai berikut (Noor, dkk, 2020):

$$u(x,t) = \sin(x)\cos(ct). \tag{2.4}$$

2. Persamaan Burger *inviscid*

Persamaan Burger *inviscid* yang akan ditentukan solusinya pada penelitian ini dinyatakan sebagai berikut:

$$u_t + uu_x = 0, \qquad 0 < x < 2\pi, \qquad t > 0 \tag{2.5}$$

dengan syarat awal

$$u(x,0) = \sin(x) \tag{2.6}$$

Berdasarkan penelitian yang dilakukan oleh Ihsan, dkk memperoleh solusi analitik Persamaan Burger *inviscid* dengan syarat awal (2.6) yaitu sebagai berikut (Ihsan, dkk, 2021):

$$u(x,t) = (\sin(x))(kt^2 + 1). \tag{2.7}$$

dengan, *k* adalah konstanta.

3. Persamaan Panas 1-D

Persamaan Panas 1-D yang akan yang akan ditentukan solusinya pada penelitian ini dinyatakan sebagai berikut:

$$u_t = 0{,}003u_{xx} \tag{2.8}$$

dengan syarat awal

$$u(x,0) = 50x(1-x) \tag{2.9}$$

dan syarat batas

$$\begin{aligned} u(0,t) &= 0 \\ u(1,t) &= 0 \end{aligned} \tag{2.10}$$

Berdasarkan penelitian yang dilakukan oleh Lebl memperoleh solusi analitik Persamaan Panas 1-D dengan syarat awal (2.9) dan syarat batas (2.10) yaitu sebagai berikut (Lebl, 2019):

$$u(x,t) = \sum_{n=1\ ganjil}^{\infty} \frac{400}{\pi^3 n^3} \sin(n\pi x)\, e^{-n^2\pi^2 0{,}003t}. \tag{2.11}$$

2.3. Klasifikasi persamaan diferensial parsial

Persamaan diferensial parsial dapat diklasifikasikan sebagai berikut:

1. Ordo dari persamaan diferensial parsial

Ordo dari PDP merupakan turunan tertinggi dari turunan parsial yang ada.

Contoh 2.1:

1. $u_{tt} = c^2 u_{xx}$ merupakan PDP berordo 2, hal ini dapat dilihat dari turunan variabel tak bebas *u.*
2. $u_t + uu_x = 0$ merupakan PDP berordo 1, hal ini dapat dilihat dari turunan variabel tak bebas *u.*
3. $u_t = c^2 u_{xx}$ merupakan PDP dengan ordo 2, hal ini dapat dilihat dari turunan variabel tak bebas *u.*

2. Banyak variabel

Banyak variabel atau peubah merupakan jumlah dari banyaknya variabel bebas.

Contoh 2.2:

1. $u_{tt} = c^2 u_{xx}$ merupakan PDP dengan dua variabel yaitu ***t,*** dan ***x***.
2. $u_t + uu_x = 0$ merupakan PDP dengan dua variabel yaitu ***t,*** dan ***x***.
3. $u_t = c^2 u_{xx}$ merupakan PDP dengan dua variabel yaitu ***t,*** dan ***x.***

3. Linearitas

Berdasarkan linearitas, PDP terbagi menjadi dua yaitu:

a. Linear

Suatu PDP dikatakan linear jika variabel-variabel tak bebas dan semua turunan dari PDP tersebut muncul dalam bentuk linear, artinya PDP memenuhi 3 aksioma berikut:

a. Variabel-variabel tak bebas dan semua turunannya muncul dalam derajat satu.
b. Tidak ada perkalian antara variabel-variabel tak bebas dan atau turunannya.
c. Tidak ada fungsi transenden dari variabel-variabel tak bebas dan atau turunannya.

Contoh 2.3:

$$u_{tt} = c^2 u_{xx}$$

a. Variabel-variabel tak bebas ***u*** muncul dalam derajat satu.
b. Dapat dilihat bahwa tidak ada perkalian antara variabel-variabel tak bebas ***u*** dan atau turunannya.
c. Tidak ada fungsi transenden (trigonometri, logaritma, dan sebagainya) dari variabel-variabel tak bebas ***u.***

Karena memenuhi tiga aksioma linear maka PDP $u_{tt} = c^2 u_{xx}$ adalah PDP linear.

b. Tak linear

PDP dikatakan tak linear jika tidak memenuhi salah satu dari ketiga aksioma linear diatas.

Contoh 2.4:

$$u_t + uu_x = 0$$

a. Terdapat perkalian antara variabel tak bebas ***u*** dengan turunannya $\boldsymbol{u_x}$. hal ini tidak memenuhi aksioma ke dua dari PDP linear, maka PDP $u_t + uu_x = 0$ adalah PDP nonlinear.

4. Homogenitas

Sebuah PDP $Au_{xx} + Bu_{xy} + Cu_{yy} + Du_x + Eu_y + Fu = G$ disebut homogen jika:

$$G = 0 \tag{2.12}$$

Contoh 2.5:

1. PDP homogen:

$$u_{tt} - c^2 u_{xx} = 0$$
$$u_t + uu_x = 0$$

Kedua persamaan diatas merupakan PDP homogen, karena $G = 0$

2. PDP tak homogen:

$$u_{xx} + u_x + u_t + u = f(x,t), \qquad f(x,t) \neq 0$$

Merupakan PDP tak homogen, karena $G = f(x,t) \neq 0$.

5. Jenis koefisien

Koefisien PDP dibedakan menjadi dua, yaitu:

a. Koefisien variabel, dimana koefisien PDP berupa variabel tak bebasnya berupa variabel x, y dan sebagainya.
b. Koefisien konstanta, dimana koefisien PDP berupa konstanta atau bilangan bulat.

6. Ketiga tipe dasar persamaan linear

PDP berordo dua dengan dua variabel yang banyak terdapat dalam aplikasi mempunyai bentuk persamaan.

$$Au_{xx} + Bu_{xy} + Cu_{yy} + Du_x + Eu_y + Fu = G \tag{2.13}$$

dengan ***A, B, C, D, E, F,*** dan ***G*** adalah merupakan konstan ataupun fungsi *x* dan *y.* berdasarkan kekoefisienannya PDP di bedakan menjadi:

1. Koefisien memenuhi persamaan $B^2 - 4AC = 0$, termasuk kelas parabolik.
2. Koefisien memenuhi persamaan $B^2 - 4AC > 0$, termasuk kelas hiperbolik.
3. Koefisien memenuhi persamaan $B^2 - 4AC < 0$, termasuk kelas eliptik.

Contoh 2.6:

1. $u_{tt} = c^2 u_{xx}$

   Dari PDP diatas dapat dilihat bahwa nilai dari $A = -c^2,\ B = 0,\ dan\ C = 1,$ maka $0^2 - 4(-c^2)(1) = 4c^2 > 0, \text{untuk setiap } c \in \mathbb{R}.$

   Jadi PDP $u_{tt} = c^2 u_{xx}$ termasuk kelas hiperbolik.

2. $u_t + uu_x = 0$

   Dari PDP diatas dapat dilihat bahwa nilai dari $A = 0,\ B = 0,\ \text{dan } C = 0,$ maka $0^2 - 4(0)(0) = 0.$

   Jadi PDP $u_t + uu_x = 0$ termasuk kelas parabolik.

3. $u_t = c^2 u_{xx}$

   Dari PDP diatas dapat dilihat bahwa nilai dari $A = c^2,\ B = 0,\ dan\ C = 0,$ maka $0^2 - 4(c^2)(0) = 0, \text{untuk setiap } c^2 \in \mathbb{R}.$

   Jadi PDP $u_t = c^2 u_{xx}$ termasuk kelas parabolik.

2.4. Persoalan nilai awal dan persoalan nilai batas

Dalam persamaan diferensial parsial sering kali harus memenuhi syarat-syarat yang diberikan. Bila syarat yang diberikan melibatkan variabel ruang, maka disebut syarat batas, dan bila syarat yang diberikan melibatkan variabel waktu, maka disebut syarat awal (Kusuma, 2018). Pada penelitian ini syarat batas yang digunakan adalah syarat batas *dirichlet.*

Diferensial Persoalan Dirichlet

Persoalan dirichlet ditandai dengan keseluruhan syarat batas yang berupa fungsi yang dicari dalam persamaan parsialnya.

Contoh 2.8:

$$u_t = c^2 u_{xx} \qquad 0 < \mathrm{x} < 1, \qquad 0 < \mathrm{t} < \infty,$$

dengan syarat awal dan syarat batas

$$u(x,0) = f(x), \qquad u(0,t) = 0, \qquad u(1,t) = 0.$$

2.5. Metode *Artificial Neural Network* (ANN)

A. Sejarah *Artificial Neural Network* (ANN)

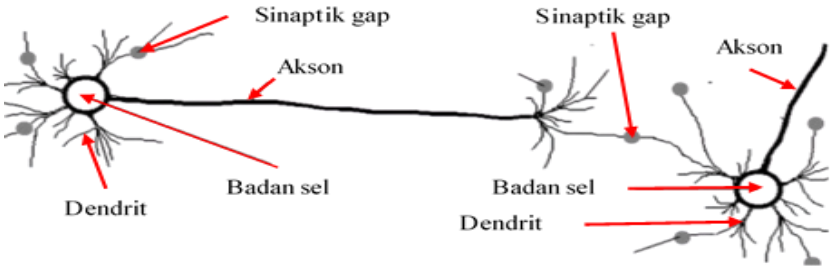
Perkembangan ilmu *Neural network* sudah ada sejak tahun 1943 ketika Warren McCulloch dan Walter Pitts memperkenalkan perhitungan model *neural*

network yang pertama kalinya. Kemudian dilanjutkan oleh Rosenblatt pada tahun 1950, dimana dia berhasil menemukan sebuah *two-layers network* yang disebut sebagai *perceptron*. *Perceptron* memungkinkan untuk pekerjaan klasifikasi pembelajaran tertentu dengan penambahan bobot pada setiap koneksi antar-*network*. Keberhasilan *perceptron* dalam pengklasifikasian pola tertentu ini tidak sepenuhnya sempurna, masih ditemukan juga beberapa keterbatasan didalamnya. *Perceptron* tidak mampu untuk menyelesaikan permasalahan XOR (exclusive-OR). Penilaian terhadap keterbatasan *neural network* ini membuat penelitian di bidang ini sempat mati selama kurang lebih 15 tahun. Namun demikian, *perceptron* berhasil menjadi sebuah dasar untuk penelitian-penelitian selanjutnya di bidang *neural network*.

B. Definisi *Artificial Neural Network* (ANN)

1. Jaringan saraf biologis

Otak manusia terdiri atas sekitar 10^{11} sel saraf (*neuron*) yang bertugas untuk memproses informasi yang masuk. Tiap sel saraf bekerja seperti prosesor dan dihubungkan dengan sel saraf lain hingga 10^{4} sinapsis. Masing-masing sel saraf tersebut saling berinteraksi sehingga mendukung kemampuan kerja otak manusia. Berikut jaringan saraf pada manusia:

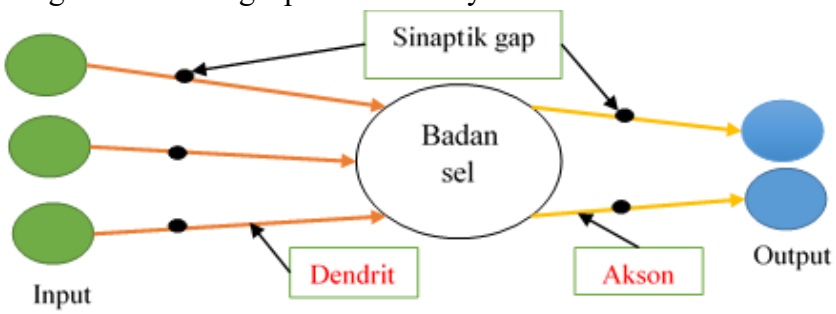


Gambar 2.1. Jaringan saraf biologis.

Komponen utama *neuron* dapat dikelompokkan menjadi 4 bagian:

1. Dendrit, bertugas menerima informasi dan jalur *input* bagi badan sel.
2. Badan sel (soma), berfungsi sebagai tempat pengolahan informasi.
3. Akson, bertugas mengirimkan impuls-impuls sinyal ke sel saraf lain dan jalur *output* bagi badan sel.
4. Sinaptik gap, berfungsi sebagai tempat menghubungkan ujung sel saraf dengan sel saraf lainnya.

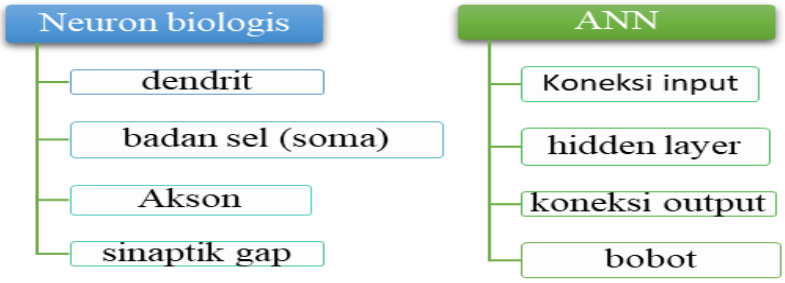
Model jaringan saraf biologis pada manusia yaitu:



Gambar 2.2 Model struktur *neuron* pada jaringan saraf biologis.

2. Jaringan saraf tiruan (*artificial neural network*)

Jaringan saraf tiruan atau *Artificial Neural Network* adalah sebuah cara pengolahan informasi yang terinspirasi dari sistem kerja saraf biologis, seperti kinerja otak yang memproses sesuai informasi. Metode ANN adalah suatu struktur baru dari sistem pengolahan informasi, yang terdiri dari sejumlah besar elemen-elemen pemrosesan yang saling berhubungan (*neuron*) dan saling bekerja sama untuk pemecahan masalah-masalah tertentu (Balaji dan Baskaran, 2013). Skema proses antara neuron biologis dan *neural network* dapat dilihat pada gambar berikut ini:



Gambar 2.3. Skema proses antara neuron biologis dan *ANN*.

Artificial neural network dibagi dalam beberapa bagian, yaitu sebagai berikut (Angel das, 2020):

a. *Input layer*

Layer ini mewakili variabel *input* ditambah dengan *neuron* bias, *neuron* bias adalah parameter tambahan yang diterapkan ke input sebelumnya, sebelum fungsi aktivasi diterapkan. Penambahan neuron bias dapat membantu meningkatkan fleksibelitas dan kemampuan adaptasi jaringan saraf. Dengan memiliki bias, neuron dapat belajar untuk menghasilkan *output* yang sesuai dalam berbagai situasi. Tanpa

neuron bias, *input* 0 untuk sebuah *network* pasti memberikan *output* 0 juga. Tentu suatu *network* dapat bekerja tanpa neuron bias, tetapi dengan menambahkan *neuron* bias, *network* yang digunakan dapat mempelajari fungsi yang lebih kompleks. Jika terdapat *i* variabel *input*, maka ukuran variabel *input* adalah $i + b_i$, dengan b_i adalah suku bias. Pada penelitian ini, *input* dinotasikan dengan *x* dan *t,* dimana *x* dan *t* adalah variabel dari persamaan differensial parsial. Informasi dari *input layer* kemudian dibawa oleh koneksi *input* ke *hidden layer* dengan besaran kekuatan hubungan tertentu (bobot).

b. Kekuatan hubungan (bobot)

Kekuatan hubungan antar sel saraf (*neuron*) yang dikenal dengan bobot-bobot sinaptik, dan digunakan untuk menyimpan pengetahuan. Inisialisasi bobot atau pemberian kekuatan hubungan antar *neuron* dapat dilakukan dengan dua acara, yaitu inisialisasi bobot secara *random* dan inisialisasi Nguyen-windrow. Pada inisialisasi secara *random* bobot diinisialisasi secara acak tanpa menggunakan faktor skala. Sedangkan pada inisialisasi Nguyen-Windrow, inisialisasi dilakukan dengan cara memodifikasi inisialisasi acak dengan menggunakan faktor skala β dengan tujuan untuk mempercepat proses pelatihan *artificial neural network* (Novikaginanto, 2012). Dalam penelitian ini, inisialisasi bobot dilakukan dengan cara inisialisasi Nguyen-windrow karena pada inisialisasi Nguyen-windrow dapat mempercepat proses pelatihan. Bobot-bobot di notasikan dengan $W_{ij,}$ dan V_i.

c. *Hidden layer*

Hidden layer merupakan *neuron* tempat semua perhitungan matematis dilakukan. Seperti halnya dengan *neuron biologis*, informasi yang dibawa oleh dendrit dari jaringan saraf lain diolah dalam badan sel yang berisi nukleus, terdapat suatu organel yang berisi material genetik, yang mengatur perkembangan dan pertumbuhan sel saraf, pada *hidden layer* juga berisi layer-layer yang mengatur pertumbuhan dan perkembangan dari *artificial neural network.* Biasanya pada jaringan saraf yang diberikan memiliki lebih dari satu layer pada lapisan tersembunyi. pada penelitian ini, jumlah *neuron* pada *hidden layer* yang digunakan adalah 10 *neuron*, 50 *neuron* dan 100 *neuron*, dengan 2 *hidden layer,* hal ini dipilih karena jumlah *input* yang terbatas. Jika terlalu banyak *hidden layer* maupun *neuron* pada *hidden layer*, maka semakin banyak parameter pada model ANN, hal ini dapat

menyebabkan *overfitting* pada data *training,* akibatnya model dapat menghasilkan prediksi yang buruk. *Hidden layer* dinotasikan dengan $h_{j,k}$. Dalam *hidden layer* terdapat dua perhitungan matematis yang digunakan, yaitu sebagai berikut:

1. *Summation function*

Dalam sebuah *neural network*, setiap neuron menerima *input* dari neuron lain dan menghasilkan *output* yang akan menjadi *input* bagi neuron selanjutnya, kemudian *input* tersebut dihitung dengan melakukan aktivasi internal yaitu dengan operasi *summation function*. operasi *summation function* dilakukan dengan menjumlahkan seluruh nilai *output* neuron dan menambahkan dengan sebuah bias. Bentuk sederhananya adalah dengan mengalikan setiap nilai input Z_j dengan nilai bobot W_{ij} dan menjumlahkannya (disebut penjumlahan berbobot ($h_{j,k}$)), kemudian dijumlahkan dengan bias-nya.

$$h_{j,k} = \sum_{i=1,j=1}^{n} W_{i,j} Z_j + b_i. \tag{2.14}$$

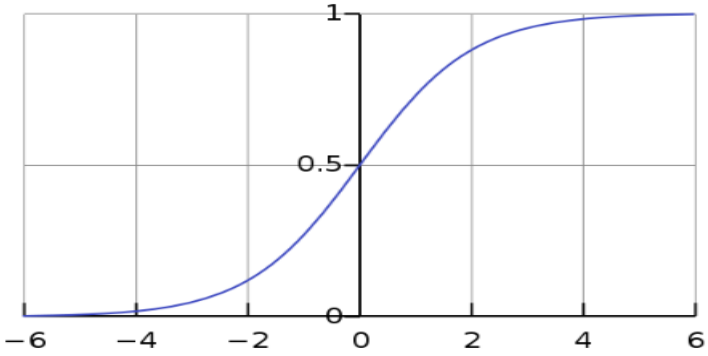
2. Fungsi aktivasi

Fungsi aktivasi merupakan fungsi yang menggambarkan hubungan antar tingkat aktivasi internal (*summation function*) yang mungkin berbentuk linier atau tak linear. Fungsi aktivasi digunakan untuk menentukan *output* pada setiap neuron.

Fungsi aktivasi sigmoid

Fungsi aktivasi sigmoid mengambil *input* bilangan rill dan menghasilkan *output* dalam rentang 0 hingga 1. Fungsi aktivasi *sigmoid* dirumuskan sebagai berikut:

$$\phi(x) = \frac{1}{1 + e^{-x}} \tag{2.15}$$



Gambar 2.4. Fungsi aktivasi *sigmoid*.

Fungsi aktivasi *sigmoid* dipilih dalam proses *training* pada kasus ini karena rentang nilainya berada diantara 0 hingga 1, hal ini dikaitkan pula dengan besar nilai *input layer* yang berada pada [0 1]. Berikut turunan dari fungsi aktivasi *sigmoid*:

$$\phi(x) = \frac{1}{1+e^{-x}}$$

Misal $u = 1, u' = 0,$

$v = 1 + e^{-x}, v' = -e^{-x}$

$$\phi(x) = \frac{u}{v}$$

$$\phi'(x) = \frac{u'v - uv'}{v^2}$$

$$\phi'(x) = \frac{0(1+e^{-x}) - 1(-e^{-x})}{(1+e^{-x})^2}$$

$$\phi'(x) = \frac{(e^{-x})}{(1+e^{-x})^2}$$

$$\phi'(x) = \left[\left(\frac{(1+e^{-x})-1}{(1+e^{-x})^2}\right)\right]$$

$$\phi'(x) = \left[\frac{1}{1+e^{-x}}\right] - \left[\left(\frac{1}{(1+e^{-x})^2}\right)\right]$$

$$\phi'^{(x)} = \left(\frac{1}{1+e^{-x}} - \left(\frac{1}{1+e^{-x}}\right)^2\right)$$

$$\phi'^{(x)} = \frac{1}{1+e^{-x}}\left(1 - \frac{1}{1+e^{-x}}\right)$$

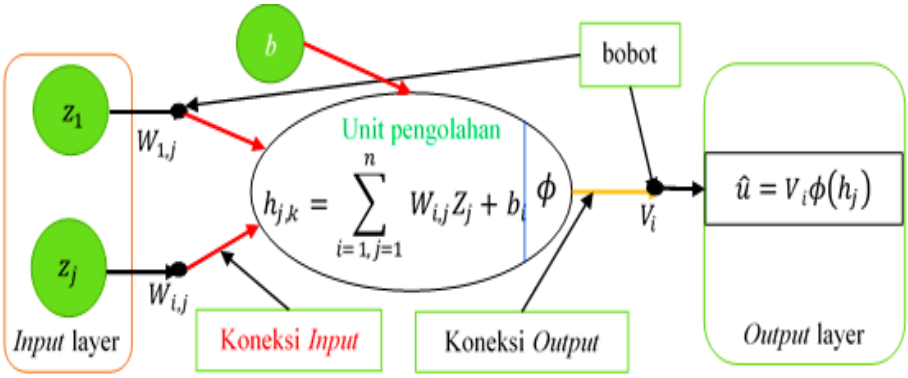
$$\phi'(x) = \phi(x)(1-\phi(x))$$

diperoleh:

$$\phi'(x) = \phi(x)(1-\phi(x)) \tag{2.16}$$

d. *Output layer*

Output layer adalah lapisan terakhir dari jaringan, dimana pada lapisan ini terdapat hasil prediksi berdasarkan variabel *input*. Cara menentukan *output layer* dari metode ini adalah mengalihkan setiap output dari *hidden layer* $\left(\phi\left(h_{i,j}\right)\right)$ dengan bobot-bobotnya $\left(V_{ij}\right)$. Pada penelitian ini *output layer* dinotasikan dengan $\hat{u}(x,t)$. Struktur *neuron* pada *artificial neural network* (ANN) disajikan seperti gambar berikut ini (Danang, dkk, 2022):



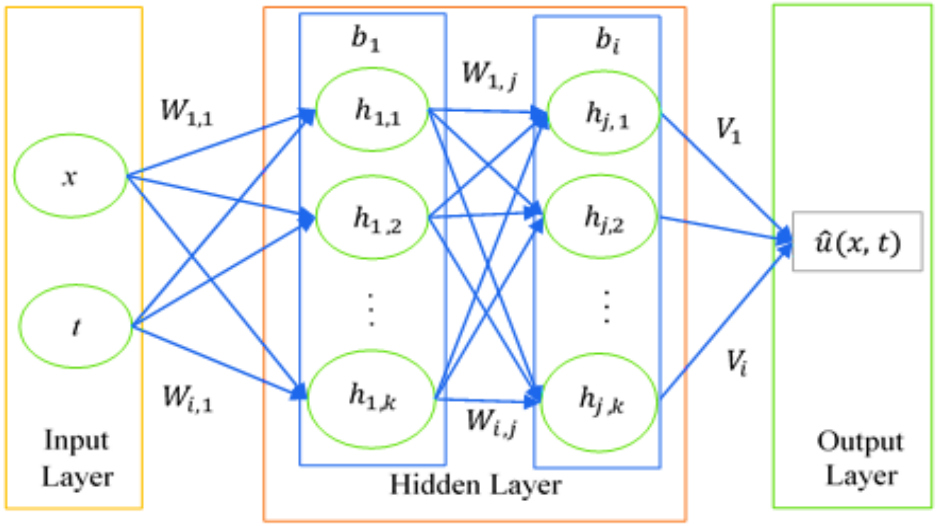
Gambar 2.5. Model struktur *neuron* pada *Artificial neural network.*

C. Arsitektur *artificial neural network*

Pada *Artificial neural network* neuron-neuron akan dikumpulkan dalam lapisan-lapisan yang disebut dengan *neuron layer*. Neuron-neuron ini akan dihubungkan mulai dari *input layer* sampai *output layer* melalui *hidden layer*. Adapun arsitektur *ANN*, yaitu sebagai berikut:

Jaringan dengan banyak *layer* (*multilayer net*)

Jaringan dengan banyak *layer* (*multilayer net*) Memiliki 1 atau lebih *layer* yang terletak di antara *input layer* dan *output layer*. Umumnya terdapat *layer* bobot-bobot yang terletak di antara dua *layer* yang bersebelahan. Jaringan dengan banyak *layer* ini dapat menyelesaikan permasalahan yang lebih sulit, dan tentu saja dengan pembelajaran yang lebih rumit. Arsitektur *artificial neural network multilayer net* dapat dilihat pada gambar berikut ini (Danang, dkk, 2022):



Gambar 2.6 Arsitektur jaringan *multilayer net*.

D. Proses pembelajaran jaringan

Peng*input*an informasi dilakukan melalui unit-unit *input*. Kemudian bobot-bobot antar koneksi dalam suatu arsitektur diberi nilai awal dan kemudian *ANN* dijalankan. Bobot-bobot ini bagi jaringan digunakan untuk belajar dan mengingat suatu informasi. Pengaturan bobot dilakukan secara terus menerus sampai diperoleh *output* yang diinginkan. Berikut metode pelatihan pada *ANN* (Idhan, 2007).

supervised learning

Pada *supervised learning* diberikan *input* variabel (x dan t) dan *output* variabel ($\hat{u}$), serta algoritma pemetaan yang dinyatakan dalam suatu fungsi $(\hat{u}) = f(x,t)$. Pendekatan pemetaan fungsi tersebut akan bekerja dengan baik jika memiliki *input* data baru (x dan t) sehingga dapat dilakukan prediksi terhadap *output* variabel ($\hat{u}$). Tujuan utama dari pembelajaran *supervised learning* adalah belajar mengasosiasikan *input* dengan *output* yang sesuai (Indarwatin, 2019). Proses pembelajaran *supervised learning* dilakukan dengan menghitung *loss function*, kemudian mengoptimalkan parameter model (bobot-bobot) untuk meminimalkan *loss function* tersebut.

E. Loss function

Suatu fungsi $\hat{u}$ dikatakan solusi suatu PDP jika memenuhi PDP yang diberikan, yaitu ruas kiri dan ruas kanan PDP tersebut harus sama. Untuk memecahkan PDP, jaringan saraf harus menemukan solusi yang memenuhi kendala yang dikenakan oleh PDP serta memenuhi syarat awal dan syarat batas dari PDP tersebut, hal ini dipenuhi dengan meminimalkan *loss function* (Remco, dkk, 2021).

Keberhasilan suatu proses pembelajaran *ANN* ditunjukkan dengan nilai *loss function* yang kecil, pada kondisi inilah *ANN* tersebut dapat digunakan (Idhan, 2007). Dalam penelitian ini, peneliti menghitung perbandingan antara solusi analitik PDP dengan solusi numerik PDP dengan algoritma BPNN menggunakan perbandingan *Root Mean Square Error* (*RMSE*).

i. *Mean Square Error* (*MSE*)

MSE merupakan nilai rata-rata kuadrat dari *error* prediksi. Pada tiap data akan dihitung kuadrat perbedaan antar solusi analitik dengan solusi numerik.

Semakin besar nilai *MSE* maka semakin buruk hasil numerik yang diperoleh (Indarwatin, 2019). *MSE* dapat dihitung dengan menggunakan persamaan:

$$MSE = \frac{1}{n}\sum_{1}^{n}(u(x,t) - \hat{u}(x,t)_{out})^2. \tag{2.17}$$

dengan,

$u(x,t)$: solusi analitik PDP.

$\hat{u}(x,t)_{out}$: solusi BPNN.

n : jumlah *input.*

ii. *Root Mean Square Error* (*RMSE*)

RMSE merupakan nilai rata-rata kuadrat dari *error* prediksi yang diakarkan. Semakin besar nilai *RMSE* maka prediksi yang dihasilkan cukup buruk (Indarwatin, 2019). *RMSE* dapat dihitung menggunakan persamaan berikut:

$$RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n}(u(x,t) - \hat{u}(x,t)_{out})^2}. \tag{2.18}$$

dengan,

$u(x,t)$: solusi analitik PDP.

$\hat{u}(x,t)_{out}$: solusi BPNN.

n : jumlah *input.*

F. Fungsi optimasi

***Stochastic Gradient Descent* (SGD)**

Stochastic gradient descent (SGD) adalah algoritma optimasi iteratif (ulang), yang berguna untuk mencari titik fungsi minimum yang dapat diturunkan. SGD selalu melakukan pembaruan bobot untuk setiap data yang dilatih (Admojo dan Sulistya, 2022). *Update* bobot dengan menggunakan SGD dapat dilakukan sebagai berikut (Haqqi dan Kusumoputro, 2022):

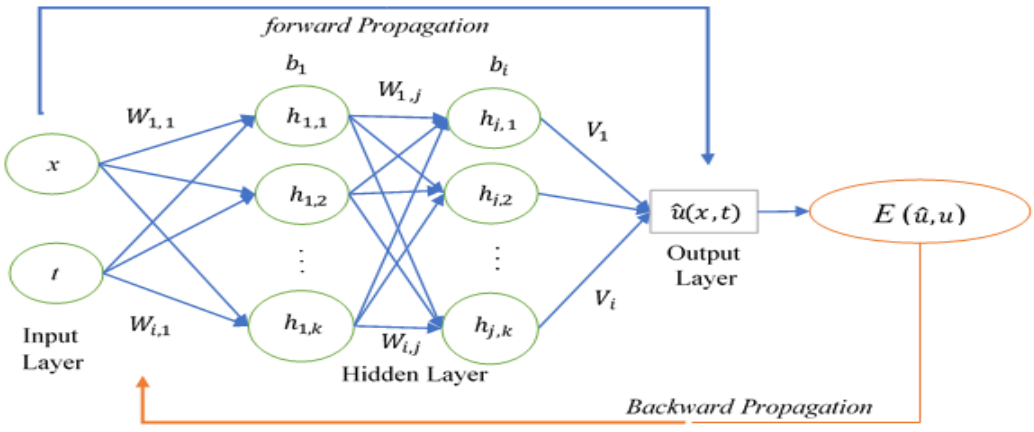
$$W_{ij}(baru) = W_{ij}(lama) - \alpha\frac{\partial E}{\partial W_{ij}}, \tag{2.19}$$

dimana α adalah *learning rate* yang dapat ditentukan secara manual dalam interval 0 hingga 1.

Learning rate (α) merupakan parameter yang digunakan untuk mengontrol seberapa besar *update* bobot dan bias pada setiap iterasi saat pelatihan BPNN. Jika α memiliki nilai yang tinggi, maka langkah penyesuaian bobot yang diambil akan lebih besar, sehingga jaringan dapat dengan cepat konvergen ke solusi yang baik. Namun, jika α terlalu besar ada resiko jaringan melompati atau melewati solusi analitiknya, dan tidak korvergen dengan baik atau bahkan menjadi tidak stabil. Sebaliknya, jika α memiliki nilai yang terlalu rendah, langkah-langkah penyesuaian bobot yang diambil akan lebih kecil, sehingga jaringan akan konvergen lebih lambat ke solusi analitiknya. Tidak ada *range* yang tepat untuk menentukan nilai α dalam backpropagation, karena nilai yang optimal dapat bervariasi tergantung pada arsitektur jaringan yang digunakan (Kholis dan Rofii, 2020). Pada penelitian ini dipilih nilai α = 0.01, hal ini dilakukan untuk menghindari terjadinya penyesuaian bobot yang tidak konvergen ataupun lebih lambat konvergen ke solusi analitiknya.

2.6. Algoritma *Backpropagation*

Algoritma *backpropagation* pertama kali dirumuskan oleh Werbos dan dipopulerkan oleh Rumelhart dan McClelland untuk dipakai pada *ANN*. *Backpropagation* adalah algoritma pembelajaran untuk memperkecil tingkat *loss function* dengan cara menyesuaikan bobotnya berdasarkan *output* yang diinginkan. *Backpropagation* merupakan sebuah metode sistematik untuk pelatihan *multilayer artificial neural network*, karena memiliki tiga *layer* dalam proses pelatihannya, yaitu *input layer*, *hidden layer* dan *output layer* (Novikaginanto, 2012). Algoritma *backpropagation* terdiri dari tiga proses yaitu *forward propagation, backward propagation* dan perubahan bobot. Berikut ini arsitektur BPNN:



Gambar 2.7 Arsitektur jaringan BPNN.

Langkah-langkah algoritma *Backpropagation Neural Network* yaitu sebagai berikut:

Langkah 0: langkah ini merupakan langkah inisialisasi bobot dengan menggunakan metode inisialisasi Nguyen-Windrow. Algoritma inisialisasi bobot dengan menggunakan Nguyen-Widrow adalah sebagai berikut:

1. Menentukan skala β,

$$\beta = 0{,}7(p)^{\frac{1}{n}}. \tag{2.20}$$

2. Inisialisasi bobot W_{ij} dan V_i secara random, dengan nilai inialisasi $W_{ij}{}'$ dan $V_{ij}{}'$adalah $-0{,}5 \leq W_{ij}{}' \leq 0{,}5$ dan $-0{,}5 \leq V_i{}' \leq 0{,}5$. Inisialisasi bobot secara acak dari rentang -0,5 hingga 0,5 biasanya digunakan karena jika kita menginisialisasi bobot dengan nilai yang terlalu terlalu kecil, maka bisa terjadi *vanishing gradient* dan sebaliknya, jika inisialisasi bobot terlalu besar, maka akan terjadi *exploding gradient*. *Vanishing gradient* adalah hilangnya gradien pada proses *backpropagation*, hal ini terjadi ketika gradien yang dihitung saat proses *backpropagation* menjadi sangat kecil sehingga tidak ada perubahan signifikan pada bobot saat pelatihan dan sebaliknya *exploding gradient* adalah meledaknya gradien pada saat perhitungan *backpropagation*. Hal ini terjadi ketika saat perhitungan gradien pada *backpropagation* menjadi sangat besar sehingga bobot mengalami perubahan yang sangat besar dan jaringan gagal belajar. Sehingga dengan menginisialisasi bobot secara acak dari rentang -0,5 hingga 0,5 kita dapat menghindari kedua masalah tersebut (novakaginanto, 2012).
3. Menghitung besarnya *magnitude* bobot W_{ij} dan V_{ij}.

$$\begin{aligned} \|W_{ij}{}'\| &= \sqrt{\sum_{j=1}^{n} (W_{ij}{}')^2}. \\ \|V_i{}'\| &= \sqrt{\sum_{j=1}^{n} (V_i{}')^2}. \end{aligned} \tag{2.21}$$

4. Meng*update* bobot W_{ij} dan V_i,

$$W_{ij} = \frac{\beta W_{ij}'}{\|W_{ij}'\|}.$$
$$V_i = \frac{\beta V_i'}{\|V_i'\|}. \tag{2.22}$$

5. Mengatur nilai bias, $-\beta \leq b_i \leq \beta$. (2.23)

Langkah 1: Menentukan nilai *epoch*, *epoch* merupakan parameter yang menentukan berapa kali algoritma *backpropagation* bekerja melewati seluruh langkah, baik langkah *forward propagation* maupun *backward propagation.*

Langkah 2: Untuk masing-masing pasangan data *training* (data *training* yang dimaksud adalah nilai *input x* dan *t* yang telah ditentukan terlebih dahulu) lakukan langkah 3-9.

Langkah 3-5 merupakan langkah propagasi maju (*forward propagation*) (Danang, dkk, 2022).

Langkah 3:

Masing-masing unit *input* x dan t menerima sinyal *input*, dan sinyal tersebut disebarkan ke unit-unit bagian berikutnya (unit-unit *hidden layer*).

Langkah 4:

Masing-masing unit *hidden layer* dikalikan dengan masing-masing nilai bobotnya dan dijumlahkan serta ditambahkan dengan biasnya:

$$h_{j,k} = \sum_{i=1,j=1}^{n,m} \left(W_{ij} Z_j\right) + b_i. \tag{2.24}$$

Kemudian menghitung *output hidden layer* sesuai dengan fungsi aktivasi yang digunakan

$$\phi\left(h_{j,k}\right) = \frac{1}{1 + e^{-h_{j,k}}}. \tag{2.25}$$

Kemudian mengirim sinyal dari Persamaan 2.11 ke unit *output layer.*

Langkah 5:

Untuk memperoleh solusi dari PDP $(\hat{u}(x,t))$, masing-masing *output* dari unit *hidden layer* $\left(h_{1,j}\right)$ dikalikan dengan masing-masing bobotnya (bobot $V_{i,j}$) dan dijumlahkan.

$$\hat{u}(x,t)_{in} = \sum_{i=1}^{n} \left(V_i \phi\left(h_{j,k}\right)\right). \tag{2.26}$$

$$\hat{u}(x,t)_{out} = \phi(\hat{u}(x,t)_{out}) = \frac{1}{1+e^{-\hat{u}(x,t)_{out}}}. \tag{2.27}$$

Langkah 6 dan langkah 7 merupakan langkah propagasi mundur (*Backward propagation*) (Avrutskiy, 2017).

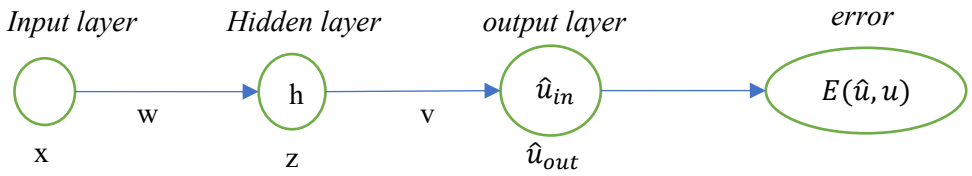
Langkah 6:

Langkah selanjutnya adalah menghitung nilai *loss function* dari PDP, yang nantinya akan digunakan untuk meng-*update* nilai bobot.

Berdasarkan Persamaan (2.17) diatas diperoleh *loss function* sebagai berikut:

$$E(\hat{u},u) = RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n}(u(x,t) - \hat{u}(x,t)_{out})^2}. \tag{2.28}$$

Selanjutnya menghitung turunan parsial *error* jaringan (E) terhadap masing-masing bobot dengan cara *chain rule*, yaitu sebagai berikut:



$h = x.w$

$z = \phi(h)$

$\hat{u}_{in} = v.z$

$\hat{u}_{out} = \phi(\hat{u}_{in})$

sehingga diperoleh

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial \hat{u}_{out}} \frac{\partial \hat{u}_{out}}{\partial \hat{u}_{in}} \frac{\partial \hat{u}_{in}}{\partial v} \frac{\partial v}{\partial z} \frac{\partial z}{\partial h} \frac{\partial h}{\partial w} \tag{2.29}$$

Menghitung turunan parsial *error* jaringan (E) terhadap masing-masing bobot V_i (untuk memperbaiki V_i), yaitu sebagai berikut:

$$\Delta V_i = \frac{\partial E}{\partial V_i} = \frac{\partial E}{\partial \hat{u}_{out}} \frac{\partial \hat{u}_{out}}{\partial \hat{u}_{in}} \frac{\partial \hat{u}_{in}}{\partial V_i}. \tag{2.30}$$

Langkah 7:

Menentukan turunan parsial *error* jaringan (*E*) terhadap masing-masing *hidden layer* $(h_{j,k})$, berdasarkan Persamaan (2.29) diperoleh:

$$\Delta h_{j,k} = \frac{\partial E}{\partial h_{j,k}} = \frac{\partial E}{\partial \hat{u}} \frac{\partial \hat{u}}{\partial \phi(h_{j,k})} \frac{\partial \phi(h_{j,k})}{\partial h_{j,k}} \tag{2.31}$$

kemudian menghitung turunan parsial *error* jaringan (*E*) terhadap masing-masing bobot $W_{i,j}$ (untuk memperbaiki $W_{i,j}$), berdasarkan Persamaan (2.29) diperoleh:

$$\Delta W_{i,j} = \frac{\partial E}{\partial W_{ij}} = \frac{\partial E}{\partial \hat{u}} \frac{\partial \hat{u}}{\partial \phi(h_{j,k})} \frac{\partial \phi(h_{j,k})}{\partial h_{j,k}} \frac{\partial h_{j,k}}{\partial W_{i,j}}. \tag{2.32}$$

dan menghitung turunan parsial *error* jaringan (*E*) terhadap nilai bias (untuk memperbaiki b_1:

$$\Delta b_i = \frac{\partial E}{\partial b_i} = \frac{\partial E}{\partial \hat{u}} \frac{\partial \hat{u}}{\partial \phi(h_{j,k})} \frac{\partial \phi(h_{j,k})}{\partial h_{j,k}} \frac{\partial h_{j,k}}{\partial b_i}. \tag{2.33}.$$

Langkah 8 merupakan langkah *update bobot.*

Langkah 8:

Masing-masing bobot diperbaiki dengan menggunakan SGD, yaitu sebagai berikut:

$$W_{i,j}(baru) = W_{i,j}(lama) - \alpha \frac{\partial E}{\partial W_{i,j}}. \tag{2.34}$$

Masing-masing bobot di unit *output layer* diperbaiki

$$V_i(baru) = V_i(lama) - \alpha \frac{\partial E}{\partial V_i}. \tag{2.35}$$

Masing-masing nilai bias diperbaiki

$$b_i(baru) = b_i(lama) - \alpha \frac{\partial E}{\partial b_i}. \tag{2.36}$$

Langkah 9:

Uji kondisi pemberhentian (akhir iterasi), yaitu *epoch* = maksimal *epoch.*

Setelah mendapatkan perubahan bobot dan bias, Langkah selanjutnya adalah *testing*. Pada langkah ini *testing* dilakukan dengan mengulangi langkah *forward propagation* dan kemudian memperoleh solusi numerik PDP.

Contoh 2.9.

Tentukan solusi numerik dari persamaan $u_t = 4u_{xx}$ dengan menggunakan metode BPNN ($\alpha = 0.01$), dengan $u(x,t) = sin\ x\ e^{-4t}$

Jawab:

Langkah 0: Inisialisasi bobot dengan menggunakan inisialisasi Nguyen-Widrow yaitu sebagai berikut:

1. Menentukan skala β,

   Diketahui variabel *input* yaitu *x* dan *t*, maka $n = 2$, dan jumlah hidden layer yang digunakan adalah 1, maka $p = 1$

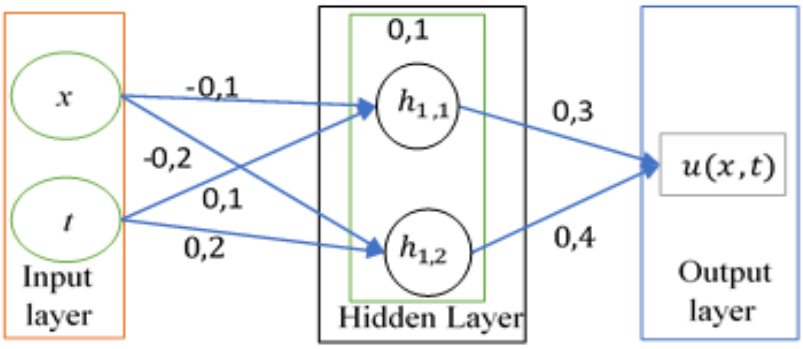
$$\beta = 0{,}7(p)^{\frac{1}{n}}$$

$$\beta = 0{,}7(1)^{\frac{1}{2}} = 0{,}7.$$

2. Inisialisasi bobot W_{ij} dan V_{ij} $i = 1{,}2$, dan $j = 1$ secara *random*, dengan nilai inialisasi $W_{ij}{}'$ dan $V_{ij}{}'$adalah $-0{,}5 \le W_{ij}{}' \le 0{,}5$ dan $-0{,}5 \le V_{ij}{}' \le 0{,}5$. diperoleh:

$$W_{11}{}' = -0{,}1 \qquad W_{22}{}' = 0{,}2 \qquad V_{11}{}' = 0{,}3$$
$$W_{12}{}' = -0{,}2 \qquad W_{21}{}' = 0{,}1 \qquad V_{12}{}' = 0{,}4.$$

   Jika dipilih *b* =0,1 maka diperoleh arsitektur sebagai berikut:



Gambar 2.8 Arsitektur jaringan dengan bobot *random*.

3. Menghitung besarnya *magnitude* bobot $W_{ij}{}'$,

$$\|W_{1j}{}'\| = \sqrt{\sum_{j=1}^{4} (W_{1j}{}')^2}$$

$$\|W_{1j}{}'\| = \sqrt{W_{11}{}'^2 + W_{12}{}'^2 + W_{21}{}'^2 + W_{22}{}'^2}$$

$$\|W_{1j}'\| = \sqrt{(-0{,}1)^2 + (-0{,}2)^2 + (0{,}1)^2 + (0{,}2)^2}$$

$$\|W_{1j}'\| = 0{,}316.$$

$$\|V_{1j}'\| = \sqrt{\sum_{j=1}^{2} (V_{1j}')^2}$$

$$\|V_{1j}'\| = \sqrt{(V_{11}')^2 + (V_{12}')^2}$$

$$\|V_{1j}'\| = \sqrt{(0{,}3)^2 + (0{,}4)^2}$$

$$\|V_{1j}'\| = 0{,}5.$$

4. Mengupdate bobot W_{ij}, dan $V_{1,j}$

$$W_{ij} = \frac{\beta W_{ij}'}{\|W_{ij}'\|}$$

$$W_{x,1} = \frac{\beta W_{11}'}{\|W_{ij}'\|} = \frac{0{,}7(-0{,}1)}{0{,}316} = -0{,}22 \qquad V_{1,1} = \frac{\beta V_{11}'}{\|V_{1j}'\|} = \frac{0{,}7(0{,}3)}{0{,}5} = 0{,}42$$

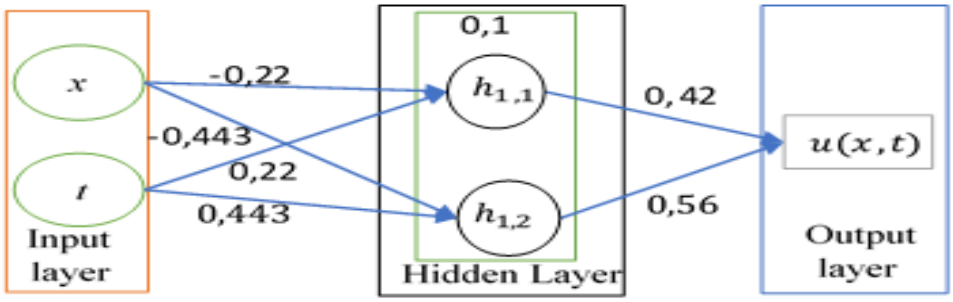
$$W_{x,2} = \frac{\beta W_{12}'}{\|W_{ij}'\|} = \frac{0{,}7(-0{,}2)}{0{,}316} = -0{,}443 \qquad V_{1,2} = \frac{\beta V_{21}'}{\|V_{1j}'\|} = \frac{0{,}7(0{,}4)}{0{,}5} = 0{,}56$$

$$W_{t,1} = \frac{\beta W_{21}'}{\|W_{ij}'\|} = \frac{0{,}7(0{,}1)}{0{,}316} = 0{,}22 \qquad W_{t,2} = \frac{\beta W_{22}'}{\|W_{ij}'\|} = \frac{0{,}7(0{,}2)}{0{,}316} = 0{,}443.$$

5. Mengatur nilai bias, $-\beta \leq b \leq \beta$

$$-0{,}5 \leq b \leq 0{,}5.$$

Jika dipilih $b = 0{,}1$, maka diperoleh arsitektur berikut ini:



Gambar 2.9. Arsitektur jaringan dengan bobot Nguyen-Windraw

Selanjutnya pilih *epoch* = 1, dan kemudian menentukan kondisi penghentian berupa maksimum *max epoch*.

Langkah 1: jika *epoch* ≠ *max epoch,* iterasi dilanjutkan.

Langkah 2: Untuk masing-masing pasangan data *traning* lakukan langkah 3-9.

Langkah 3-5 merupakan langkah propagasi maju (*forward propagation*)

Langkah 3: Masing-masing unit input ($x, \text{dan } t$) menerima sinyal input dan sinyal tersebut disebarkan ke unit-unit bagian berikutnya (*unit-unit hidden layer*).

Langkah 4: Masing-masing unit di *hidden layer* dikalikan dengan masing-masing bobotnya dan dijumlahkan serta ditambahkan dengan biasnya.

$$h_{1,j} = \sum_{i=1}^{n} (W_{xi}x + W_{ti}t) \; + \; b_1$$

$$h_{1,1} = (W_{x,1})x + (W_{t,1})t \; + \; b_1$$

$$h_{1,1} = (-0{,}22)x + (0{,}22)t \; + \; 0{,}1. \tag{1}$$

$$h_{1,2} = (W_{x,2})x + (W_{t,2})t \; + \; b_1$$

$$h_{1,2} = (-0{,}443)x + (0{,}443)t \; + 0{,}1. \tag{2}$$

Kemudian menghitung *output* $h_{1,i}$ sesuai dengan fungsi aktivasi yang digunakan:

$$\phi(h_{1,j}) = \frac{1}{1 + e^{-h_{1,i}}}$$

$$\phi(h_{1,1}) = \frac{1}{1 + e^{-(h_{1,1})}}. \tag{3}$$

$$\phi(h_{1,2}) = \frac{1}{1 + e^{-(h_{1,2})}}. \tag{4}$$

Kemudian mengirim sinyal tersebut ke unit *output layer.*

Langkah 5: untuk memperoleh solusi dari PDP, masing-masing *output* dari unit *hidden layer* $\left(h_{1,j}\right)$ dikalikan dengan masing-masing bobotnya (bobot $V_{i,j}$) dan dijumlahkan:

$$\hat{u}(x,t)_{in} = \sum_{j=1}^{2} V_{1,j}\, \phi(h_{1,j})$$

$$\hat{u}(x,t)_{in} = V_{1,1}\left(\frac{1}{1 + e^{-(h_{1,1})}}\right) + V_{1,2}\left(\frac{1}{1 + e^{-(h_{1,2})}}\right). \tag{5}$$

$$\hat{u}(x,t)_{in} = 0{,}42\left(\frac{1}{1 + e^{-(h_{1,1})}}\right) + 0{,}56\left(\frac{1}{1 + e^{-(h_{1,2})}}\right). \tag{6}$$

Sehingga diperoleh:

$$\hat{u}(x,t)_{out} = \; \phi(\hat{u}(x,t)_{in})$$

$$\hat{u}(x,t)_{out} = \frac{1}{1+e^{-\hat{u}(x,t)_{in}}}$$

$$\hat{u}(x,t)_{out} = \frac{1}{1+e^{-\left(0{,}42\left(\frac{1}{1+e^{-(h_{1,1})}}\right)+0{,}56\left(\frac{1}{1+e^{-(h_{1,2})}}\right)\right)}}. \quad (7)$$

$$\phi'(\hat{u}(x,t)_{in}) = \hat{u}(x,t)_{in}(1-\hat{u}(x,t)_{in}) \quad (8)$$

Kemudian menentukan RMSE dari model, yaitu sebagai berikut:

$$E(\hat{u},u) = \text{RMSE} = \sqrt{\frac{1}{n}\sum_{1}^{n}(u(x,t)-\hat{u}(x,t)_{out})^2}$$

$$E(\hat{u},u) = \sqrt{\frac{1}{n}\sum_{1}^{n}\left((sin\, x\, e^{-4t})-\hat{u}(x,t)_{out}\right)^2} \quad (9)$$

Misalkan data *training* $x = 0.2,\ dan\ t = 0.3$.

Berdasarkan Persamaan (1) diperoleh:

$$h_{1,1} = (-0{,}22)(0{,}2) + (0{,}22)(0{,}3) + 0{,}1 = 0{,}122. \quad (10)$$

Berdasarkan Persamaan (2) diperoleh:

$$h_{1,2} = (-0{,}443)(0{,}2) + (0{,}443)(0{,}3) + 0{,}1 = 0{,}1443. \quad (11)$$

Berdasarkan Persamaan (3) diperoleh:

$$\phi(h_{1,1}) = \frac{1}{1+e^{-(0{,}122)}} = 0{,}53. \quad (12)$$

Berdasarkan Persamaan (4) diperoleh:

$$\phi(h_{1,2}) = \frac{1}{1+e^{-(0{,}1443)}} = 0{,}536. \quad (13)$$

Berdasarkan Persamaan (6) diperoleh:

$$\hat{u}(x,t)_{in} = 0{,}42(0{,}53) + 0{,}56(0{,}536) = 0{,}522971. \quad (14)$$

Berdasarkan Persamaan (8) diperoleh:

$$\hat{u}(x,t)_{out} = 0{,}6278 \quad (15)$$

Berdasarkan Persamaan (9) diperoleh:

$$\phi'(\hat{u}(x,t)_{in}) = 0{,}2495 \quad (16)$$

Backward propagation

Langkah 6: selanjutnya menghitung nilai *loss function* dari PDP dengan menggunakan *RMSE*, yaitu sebagai berikut:

$$E = \sqrt{\frac{1}{n}\sum_{1}^{n}(u(x,t) - \hat{u}(x,t)_{out})^2}$$

Berdasarkan Persamaan (10) diperoleh:

$$E(\hat{u},u) = \sqrt{\frac{1}{n}\sum_{1}^{n}\left((sin\ x\ e^{-4t}) - (0{,}6274)\right)^2} \quad (17)$$

$$E = 0{,}40164$$

Selanjutnya menghitung turunan parsial *error* jaringan (E) terhadap masing-masing bobot V_i (untuk memperbaiki V_i) berdasarkan Persamaan (2.29):

$$\Delta V_i = \frac{\partial E}{\partial V_i} = \frac{\partial E}{\partial \hat{u}_{out}}\frac{\partial \hat{u}_{out}}{\partial \hat{u}_{in}}\frac{\partial \hat{u}_{in}}{\partial V_i}.$$

$$\Delta V_i = \left(\frac{\hat{u}(x,t)_{out} - u(x,t)}{n\sqrt{\frac{1}{n}\sum_{1}^{n}(u(x,t) - \hat{u}(x,t)_{out})^2}}\right)(\phi'(\hat{u}(x,t)_{in})\left(\phi(h_{1,j})\right)$$

$$\Delta V_1 = \left(\frac{(0{,}6274) - (sin\ x\ e^{-4t})}{2\sqrt{\frac{1}{n}\sum_{1}^{n}(sin\ x\ e^{-4t} - 0{,}6274)^2}}\right)(0{,}2495)\left(\phi(h_{1,1})\right)$$

$$\Delta V_1 = \frac{\partial E}{\partial V_i} = 0{,}0936$$

$$\Delta V_2 = \left(\frac{(0{,}6274) - (sin\ x\ e^{-4t})}{2\sqrt{\frac{1}{n}\sum_{1}^{n}(sin\ x\ e^{-4t} - 0{,}6274)^2}}\right)(0{,}2495)\left(\phi(h_{1,2})\right)$$

$$\Delta V_2 = \frac{\partial E}{\partial V_2} = 0{,}0946.$$

Langkah 7:

Menentukan turunan parsial *error* jaringan (E) terhadap masing-masing *hidden layer* $\left(h_{1,j}\right)$ berdasarkan Persamaan (2.30):

$$\Delta h_{j,k} = \frac{\partial E}{\partial h_{j,k}} = \frac{\partial E}{\partial \hat{u}} \frac{\partial \hat{u}}{\partial \phi(h_{j,k})} \frac{\partial \phi(h_{j,k})}{\partial h_{j,k}}$$

$$\Delta h_{j,k} = \left(\frac{\hat{u}(x,t)_{out} - u(x,t)}{n\sqrt{\frac{1}{n}\sum_1^n (u(x,t) - \hat{u}(x,t)_{out})^2}} \right) (V_i) \left(\phi'(h_{j,k}) \right)$$

$$\Delta h_{1,1} = \left(\frac{(0{,}6274) - (sin\ x\ e^{-4t})}{2\sqrt{\frac{1}{n}\sum_1^n (sin\ x\ e^{-4t} - 0{,}6274)^2}} \right) (V_1) \left(\phi'(h_{1,1}) \right)$$

$$\Delta h_{1,1} = \frac{\partial E}{\partial h_{1,1}} = 0{,}074$$

$$\Delta h_{1,2} = \left(\frac{(0{,}6274) - (sin\ x\ e^{-4t})}{2\sqrt{\frac{1}{n}\sum_1^n (sin\ x\ e^{-4t} - 0{,}6274)^2}} \right) (V_2) \left(\phi'(h_{1,2}) \right)$$

$$\Delta h_{1,2} = \frac{\partial E}{\partial h_{1,2}} = 0{,}070.$$

Selanjutnya menghitung turunan parsial *error* jaringan (E) terhadap masing-masing bobot $w_{1,j}$ (untuk memperbaiki $w_{1,j}$) berdasarkan Persamaan (2.31):

$$\Delta W_{x,j} = \frac{\partial E}{\partial W_{x,j}} = \frac{\partial E}{\partial \hat{u}} \frac{\partial \hat{u}}{\partial \phi(h_{j,k})} \frac{\partial \phi(h_{j,k})}{\partial h_{j,k}} \frac{\partial h_{j,k}}{\partial W_{x,j}}$$

$$\Delta W_{x,j} = \frac{\partial E}{\partial W_{x,j}} = \frac{\partial E}{\partial h_{j,k}} \frac{\partial h_{j,k}}{\partial W_{x,j}}$$

$$\Delta W_{x,1} = \frac{\partial E}{\partial W_{x,1}} = \frac{\partial E}{\partial h_{1,1}} \frac{\partial h_{1,1}}{\partial W_{x,1}}$$

$$\Delta W_{x,1} = 0{,}074(x)$$

$$\Delta W_{x,1} = 0{,}074(0{,}2) = 0{,}0148$$

$$\Delta W_{x,2} = \frac{\partial E}{\partial W_{x,2}} = \frac{\partial E}{\partial h_{1,2}} \frac{\partial h_{1,2}}{\partial W_{x,2}}$$

$$\Delta W_{x,2} = 0{,}070(x)$$

$$\Delta W_{x,2} = 0{,}070(0{,}2) = 0{,}0141$$

$$\Delta W_{t,1} = \frac{\partial E}{\partial W_{t,1}} = \frac{\partial E}{\partial h_{1,1}} \frac{\partial h_{1,1}}{\partial W_{t,1}}$$

$$\Delta W_{t,1} = \frac{\partial E}{\partial h_{1,1}} (t)$$

$$\Delta W_{t,1} = 0{,}074(0{,}3) = 0{,}0222$$

$$\Delta W_{t,2} = \frac{\partial E}{\partial W_{t,2}} = \frac{\partial E}{\partial h_{1,2}} \frac{\partial h_{1,2}}{\partial W_{t,2}}$$

$$\Delta W_{t,2} = \frac{\partial E}{\partial h_{1,2}}(t)$$

$$\Delta W_{t,2} = 0{,}070(0{,}3) = 0{,}0211$$

Menghitung turunan parsial *error* jaringan (E) terhadap bias (untuk memperbaiki b_1) berdasarkan Persamaan (2.32):

$$\Delta b_i = \frac{\partial E}{\partial b_i} = \frac{\partial E}{\partial \hat{u}} \frac{\partial \hat{u}}{\partial \phi(h_{j,k})} \frac{\partial \phi(h_{j,k})}{\partial h_{j,k}} \frac{\partial h_{j,k}}{\partial b_i}.$$

$$\Delta b_i = \frac{\partial E}{\partial h_{j,k}} \frac{\partial h_{j,k}}{\partial b_i}$$

$$\Delta b_1 = \frac{\partial E}{\partial h_{1,1}} (1) = 0{,}74.$$

Langkah 8 merupakan langkah *update* bobot

Langkah 8:

Masing-masing bobot di unit *input* diperbaiki, dimana $\alpha = 0{,}01$

$$W_{i,j}(baru) = W_{i,j}(lama) - \alpha \frac{\partial E_\tau}{\partial W_{i,j}}$$

$$W_{x,1}(baru) = W_{x,1}(lama) - \alpha \frac{\partial E_\tau}{\partial W_{x,1}}$$

$$W_{x,1}(baru) = (-0{,}221) - 0{,}01(0{,}0144)$$

$$W_{x,1}(baru) = -0{,}2215$$

$$W_{x,2}(baru) = W_{x,2}(lama) - \alpha \frac{\partial E_\tau}{\partial W_{x,2}}$$

$$W_{x,2}(baru) = (-0{,}433) - 0{,}01(0{,}0137)$$

$$W_{x,2}(baru) = -0{,}44286$$

$$W_{t,1}(baru) = W_{t,1}(lama) - \alpha \frac{\partial E_\tau}{\partial W_{t,1}}$$

$$W_{t,1}(baru) = (0{,}221) - 0{,}01(0{,}0216) = 0{,}2211$$

$$W_{t,2}(baru) = W_{t,2}(lama) - \alpha \frac{\partial E_\tau}{\partial W_{t,2}}$$

$$W_{t,2}(baru) = (0{,}443) - 0{,}01(0{,}0205) = 0{,}4425.$$

Bias diperbaiki

$$b_1(baru) = b_1(lama) - \alpha \frac{\partial E_\tau}{\partial b_1}$$

$$b_1(baru) = (0{,}1) - 0{,}01(0{,}074) = 0{,}0993.$$

Masing-masing bobot di unit *output* diperbaiki

$$V_{1,j}(baru) = V_{1,j}(lama) - \alpha \frac{\partial E_\tau}{\partial V_{1,j}}$$

$$V_{1,1}(baru) = V_{1,1}(lama) - \alpha \frac{\partial E_\tau}{\partial V_{1,1}}$$

$$V_{1,1}(baru) = (0{,}42) - 0{,}01(0{,}0936) = 0{,}419$$

$$V_{1,2}(baru) = V_{1,2}(lama) - \alpha \frac{\partial E_\tau}{\partial V_{1,2}}$$

$$V_{1,2}(baru) = (0{,}56) - 0{,}01(0{,}0946) = 0{,}5591.$$

Langkah 9:

Uji kondisi pemberhentian (akhir iterasi)

Selanjutnya menghitung kembali dengan cara *forward propagation* berdasarkan perubahan bobot

$$h_{1,1} = (-0{,}2215)(0{,}2) + (0{,}2211)(0{,}3) + 0{,}0993 = 0{,}1213$$

$$h_{1,2} = (-0{,}44286)(0{,}2) + (0{,}4425)(0{,}3) + 0{,}0993 = 0{,}14341$$

$$\phi(h_{1,1}) = \frac{1}{1 + e^{-[0{,}121323]}} = 0{,}5303$$

$$\phi(h_{1,2}) = \frac{1}{1 + e^{-[0{,}14347]}} = 0{,}5358$$

$$\hat{u}(x,t) = 0{,}62756$$

$$E = RMSE = 0{,}40144.$$

Berdasarkan perhitungan yang dilakukan, dapat dilihat bahwa nilai *loss function* dari perhitungan *forward propagation* (sebelum *update bobot*) diperoleh $E = 0{,}40164$, dan setelah melakukan proses *backward propagation* (setelah *update bobot*) terjadi penurunan *loss function* yaitu $E = 0{,}40144$. karena *loss function* awal < *loss function akhir* maka dapat disimpulkan bahwa iterasi berhasil. Jadi solusi BPNN untuk persamaan $u_t = 4u_{xx}$ dan solusi analitik $u(x,t) = \sin x\, e^{-4t}$, dengan nilai x = 0,2 dan t =0,3 adalah $\hat{u} = 0{,}62757$.