JATIKOM: Jurnal Aplikasi dan Teori Ilmu Komputer | ISSN: 2615-577X (Online)
https://ejournal.upi.edu/index.php/JATIKOM

Implementasi Metode Machine Learning menggunakan Algoritma Evolving Artificial Neural Network pada Kasus Prediksi Diagnosis Diabetes

Implementation of Machine Learning Method using Evolving Artificial Neural Network Algorithm in Diabetes Diagnosis Prediction Case

Yudi Ahmad Hambali [*1], Rani Megasari[2], Resky Ramadhandi Santoso[3]

Prodi Studi Ilmu Komputer Departemen Pendidikan Ilmu Komputer Fakultas Pendidikan Matematika dan Ilmu Pengetahuan Alam Universitas Pendidikan Indonesia
Jl. Dr. Setiabudhi No. 229 Bandung 40154 Jawa Barat – Indonesia
[1]yudi.a.hambali@upi.edu, [2]megasari@upi.edu, [3]reskyramadhandi@student.upi.edu

Abstract— **Diabetes mellitus is a global health problem that can affect anyone, from children, adolescents, to adults. Therefore, diabetes is one of the non-communicable diseases that has become a serious threat to global health. Since 1980, the number of diabetics worldwide has nearly doubled from 4.7% to 8.5% of the total population. The International Diabetes Federation (IDF) even estimates that the number of diabetes sufferers worldwide will reach 700 million people by 2045. In response to this condition, this study predicts diabetes diagnosis using machine learning algorithms, artificial neural network. However, there is a major problem with this algorithm, namely in determining the correct architecture. This problem can be viewed as an optimization problem, where many architectural possibilities that can occur. Therefore, to search for the right architecture to increase the accuracy of the predictions, there will be stages to use the evolution algorithm. Because this algorithm is very suitable to be applied in an optimization case. This study implements Evolving Artificial Neural Network (EANN) algorithm to predict the patient's diagnosis. It is with the hope that this study can produce higher accuracy in predicting patient diagnosis in diabetes. The data set used was Pima Indian Diabetes from the UCI Machine Learning Repository. Based on the experiments that have been carried out, the best model produced has an accuracy of 83.55%. This means that the algorithm used is quite successful in predicting diabetes diagnosis.**

Keywords*— *classification, diabetes mellitus, evolving artificial neural network, machine learning.

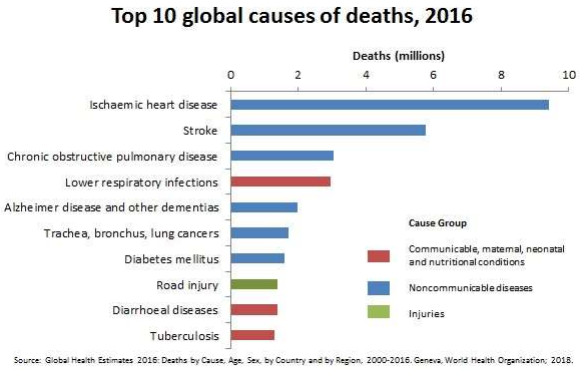
Abstrak— **Diabetes melitus merupakan masalah kesehatan global yang dapat menyerang siapa saja, mulai dari anak-anak, remaja, hingga orang dewasa. Oleh karena itu, diabetes merupakan salah satu penyakit tidak menular yang menjadi ancaman serius bagi kesehatan global. Sejak tahun 1980, jumlah penderita diabetes di seluruh dunia telah meningkat hampir dua kali lipat dari 4,7% menjadi 8,5% dari total populasi. *International Diabetes Federation* (IDF) bahkan memperkirakan jumlah penderita diabetes di seluruh dunia akan mencapai 700 juta orang pada tahun 2045. Menyikapi kondisi ini, penelitian ini memprediksi diagnosis diabetes menggunakan algoritma machine learning, jaringan syaraf tiruan. Namun, terdapat masalah utama pada algoritma ini, yaitu dalam menentukan arsitektur yang tepat. Masalah ini dapat dipandang sebagai masalah optimasi, dimana banyak kemungkinan arsitektur yang dapat terjadi. Oleh karena itu, untuk mencari arsitektur yang tepat untuk meningkatkan akurasi prediksi, maka akan dilakukan tahapan-tahapan dengan menggunakan algoritma evolusi. Karena algoritma ini sangat cocok untuk diterapkan dalam kasus optimasi. Penelitian ini mengimplementasikan algoritma Evolving Artificial Neural Network (EANN) untuk memprediksi diagnosis pasien. Dengan harapan penelitian ini dapat menghasilkan akurasi yang lebih tinggi dalam memprediksi diagnosis pasien diabetes. Data set yang digunakan adalah Pima Indian Diabetes dari UCI Machine Learning Repository. Berdasarkan percobaan yang telah dilakukan, model terbaik yang dihasilkan memiliki akurasi sebesar 83.55%. Hal ini berarti algoritma yang digunakan cukup berhasil dalam memprediksi diagnosis diabetes.**

***Kata kunci*— klasifikasi, diabetes melitus, jaringan saraf tiruan yang berkembang, machine learning.**

I. PENDAHULUAN

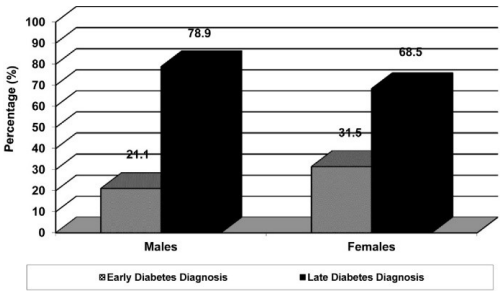
Diabetes Melitus merupakan salah satu masalah dalam dunia kesehatan yang dapat menyerang siapa saja, mulai dari anak-anak, remaja, hingga orang dewasa. Oleh karena itu, Diabetes merupakan salah satu Penyakit Tidak Menular (PTM) yang saat ini telah menjadi ancaman serius bagi kesehatan global. Dikutip dari data *World Health Organization* (WHO), 70% dari total kematian di dunia disebabkan oleh penyakit tidak menular. Bahkan hingga tahun 2016, diabetes menempati peringkat ketujuh penyebab kematian di seluruh dunia [1], seperti yang terlihat pada

Gambar 1. Fakta lain yang perlu diwaspadai adalah, posisi diabetes sebagai salah satu *silent killer* di Indonesia mungkin saja naik bila penyakit ini tidak ditangani dengan baik. Di tahun 2016, persentase kematian akibat diabetes di Indonesia mencapai 6,7% dan merupakan yang tertinggi kedua setelah Sri Lanka. Angka ini cukup tinggi, karena 2 dari 3 penderita diabetes di Indonesia tidak mengetahui bahwa dirinya mengidap diabetes. Kebanyakan dari mereka baru mengakses layanan kesehatan ketika sudah dalam kondisi yang memburuk, bahkan sudah mengalami komplikasi. Akibatnya, penyakit menjadi lebih sulit disembuhkan [2].



Gambar 1. Statistik Penyebab Kematian Global tahun 2016 [1].

Sebuah penelitian yang dilakukan di Kanada terhadap para pengidap diabetes mendapatkan hasil bahwa 74,2% dari 7101 responden survei mengatakan bahwa mereka terlambat didiagnosis. Jika dilihat berdasarkan jenis kelamin, persentase keterlambatan diagnosis untuk laki-laki sebesar 78,9%, sedangkan untuk perempuan adalah sebesar 68,5% (Gambar 2). Keterlambatan diagnosis diabetes mengacu pada ditemukannya komplikasi penyakit lain yang menyertai ketika dihasilkan diagnosis bahwa seseorang mengidap diabetes.



Gambar 2. Persentase Keterlambatan Diagnosis Diabetes pada Laki-laki dan Perempuan [3].

Lebih jauh lagi, diabetes tipe 2 khususnya dapat terjadi 9 sampai 12 tahun sebelum terdiagnosis [3]. Sejak tahun 1980, jumlah penderita diabetes di seluruh dunia telah meningkat hampir dua kali lipat dari 4,7% menjadi 8,5% dari total populasi. IDF bahkan memperkirakan jumlah penderita diabetes di seluruh dunia akan mencapai 700 juta orang pada tahun 2045 [4].

Indonesia pun menghadapi situasi ancaman diabetes serupa dengan dunia. *International Diabetes Federation* (IDF) Atlas 2017 melaporkan bahwa banyaknya jumlah penderita diabetes di Indonesia masih menunjukkan kecenderungan untuk terus meningkat. Indonesia adalah negara peringkat keenam di dunia setelah Tiongkok, India, Amerika Serikat, Brazil dan Meksiko dengan jumlah penyandang Diabetes usia 20-79 tahun sekitar 10,3 juta orang [5]. Sejalan dengan hal tersebut, Riset Kesehatan Dasar (Riskesdas) yang dilakukan oleh Kementerian Kesehatan RI memperlihatkan adanya peningkatan pada angka penderita Diabetes yang cukup signifikan, yaitu dari 6,9% di tahun 2013 menjadi 8,5% di tahun 2018, sehingga estimasi jumlah penderita di Indonesia mencapai lebih dari 16 juta orang yang kemudian berisiko terkena penyakit lain, seperti: serangan jantung, stroke, kebutaan, dan gagal ginjal bahkan dapat menyebabkan kelumpuhan dan kematian.

Inovasi-inovasi dalam pencegahan dan pengendalian serta pengobatan diabetes dinilai sangat penting untuk dilakukan. Namun, selain membentuk berbagai kebijakan untuk pengendalian dan pencegahan diabetes, sudah terdapat beberapa penelitian juga yang dilakukan di dunia ataupun di Indonesia mengenai penyakit diabetes. Penelitian-penelitian tersebut di antaranya melakukan penelitian untuk memprediksi diagnosis pasien berdasarkan data riwayat kondisi kesehatan serta hasil diagnosis pasien-pasien sebelumnya menggunakan *machine learning*. Salah satu dari berbagai macam algoritma yang dikenal juga dalam *machine learning* adalah algoritma *Artificial Neural Network* (ANN). Secara sederhana, ANN merupakan sebuah model komputasi yang terinspirasi dari ilmu biologi, yang terdiri dari elemen pemrosesan yang disebut neuron dan koneksi/jaringan di antara mereka yang disebut dengan koefisien atau bobot [6]. Beberapa poin keunggulan dari ANN dalam konteks untuk memprediksi diagnosis medis adalah tidak terlalu membutuhkan proses *training* dengan keilmuan statistika untuk dikembangkan, kemampuan untuk secara implisit mendeteksi hubungan non-linier yang kompleks antara variabel dependen dan independen, kemampuan untuk mendeteksi semua interaksi yang mungkin antara variabel prediktor, dan dapat dikembangkan dengan beberapa algoritma untuk proses *training* [7].

Namun, algoritma ANN memiliki permasalahan utama dalam hal penentuan arsitektur yang tepat. Maksud dari arsitektur di sini adalah struktur dan pembobotan antar *node* di dalam ANN. Permasalahan ini dapat dipandang sebagai permasalahan optimasi, dimana terdapat banyak sekali kemungkinan arsitektur yang bisa terjadi. Maka dari itu, untuk meningkatkan akurasi dari prediksi akan ada tahapan untuk menggunakan algoritma evolusi. Karena algoritma ini sangat cocok untuk diterapkan pada kasus optimasi [8]. Algoritma evolusi pada ANN dapat dilakukan pada tiga titik, yaitu: nilai *weight*, arsitektur, dan *learning rules*. Evolusi pada arsitektur memungkinkan ANN untuk menyesuaikan topologi mereka tanpa campur tangan manusia. Dengan demikian, hal ini memberikan pendekatan untuk menghasilkan desain arsitektur ANN secara otomatis karena bobot dan struktur koneksi ANN dapat dikembangkan. Evolusi *learning rules* dapat dianggap sebagai proses "*learning to learn*" pada ANN, di mana adaptasi *learning rules* dicapai melalui evolusi. Ini juga dapat dianggap sebagai *adaptive process of automatic discovery of novel learning rule* [9].

Dengan berbagai paparan di atas, mulai dari penelitian sejenis dengan berbagai algoritma yang digunakan, kemudian pemaparan mengenai algoritma ANN beserta kelebihan dan kekurangan serta salah satu cara untuk mengoptimasi algoritma ANN. Maka dalam penelitian ini penulis akan mengimplementasikan algoritma *Evolving Artificial Neural Network* (EANN) untuk memprediksi diagnosis pasien. Dengan harapan penelitian ini dapat menghasilkan akurasi yang lebih tinggi dalam memprediksi diagnosis pasien dalam hal penyakit diabetes.

Set data yang digunakan dalam penelitian ini menggunakan data sekunder yang berasal dari set data *Pima Indians Diabetes* yang disediakan oleh *"National Institute of Diabetes and Digestive and Kidney Diseases"* dan dikelola oleh Kaggle. Suku Pima Indian adalah sekelompok penduduk asli Amerika yang tinggal di Arizona. Mereka adalah kelompok orang yang sering kali diteliti atau dipelajari karena kecenderungan genetik mereka yang sangat tinggi terhadap diabetes. Dipercayai bahwa orang Pima Indian membawa gen yang memungkinkan mereka untuk bertahan hidup dalam menghadapi kelaparan yang lama. Gen ini memungkinkan orang Pima Indian untuk menyimpan glukosa dan karbohidrat di dalam tubuh mereka apa pun yang mereka makan, yang secara genetik hal ini menguntungkan untuk mereka yang berada di lingkungan atau tempat dimana kelaparan biasa terjadi [10]. Set data Pima Indians Diabetes ini pernah digunakan di beberapa penelitian serta buku. Dalam situs Kaggle, set data ini juga termasuk set data yang banyak digunakan para pengguna. Oleh karena itu, penelitian ini akan menggunakan set data ini sebagai bahan penelitian karena set data ini memiliki kredibilitas yang tinggi untuk digunakan dalam penelitian.

II. TINJAUAN PUSTAKA

A. Penelitian Terdahulu

I Putu Dodi Lesmana melakukan penelitian untuk diagnosis diabetes menggunakan algoritma *decision tree J48*. Dalam penelitian Lesmana tersebut data yang digunakan juga sama seperti yang digunakan penulis, yaitu set data Pima Indian Diabetes. Penelitian ini dilakukan menggunakan Weka dan menghasilkan akurasi sebesar 74,72% [11]. Ahmad Setiadi juga melakukan penelitian menggunakan set data Pima Indian Diabetes. Algoritma yang digunakan dalam penelitiannya adalah *Multi Layer Perceptron* dengan memanfaatkan *tool* SPSS Neural Network. Akurasi prediksi yang didapatkan pada penelitian ini bernilai 77,7% [12]. Terdapat pula penelitian serupa yang masih menggunakan set data yang sama yang dilakukan oleh Anik Andriani. Dalam penelitiannya, Andriani menggunakan algoritma *Decision Tree* dengan memanfaatkan *tool* Rapid Miner. Penelitian yang dilakukannya menghasilkan prediksi dengan akurasi sebesar 73,3%.

B. Diabetes Melitus

Diabetes berasal dari bahasa Yunani yang berarti "mengalirkan atau mengalihkan" (*siphon*). Sedangkan Melitus berasal dari bahasa latin yang bermakna manis atau madu. Diabetes Melitus adalah penyakit yang ditandai dengan terjadinya hiperglikemia dan gangguan metabolisme karbohidrat, lemak, dan protein yang dihubungkan dengan kekurangan secara absolut atau relatif dari kerja dan atau sekresi insulin [13]. Diabetes Melitus biasa disebut dengan *the silent killer* karena penyakit ini dapat mengenai semua organ tubuh dan menimbulkan berbagai macam keluhan. Penyakit yang akan ditimbulkan antara lain gangguan penglihatan mata, katarak, penyakit jantung, sakit ginjal, impotensi seksual, luka sulit sembuh dan membusuk, infeksi paru-paru, gangguan pembuluh darah, stroke, dan sebagainya. Tidak jarang, penderita Diabetes Melitus yang sudah parah menjalani amputasi anggota tubuh karena terjadi pembusukan.

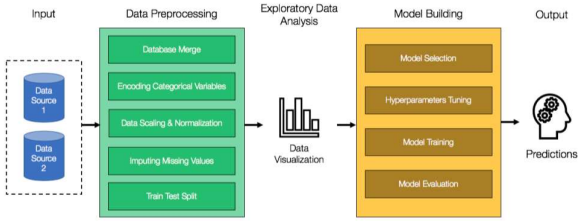
Karakteristik diagnosis dapat ditegakkan melalui tiga cara dengan melihat dari tabel 1 di bawah ini:

TABEL I. KRITERIA DIAGNOSTIK DIABETES MELITUS [14]

Kriteria Diagnostik Diabetes Melitus
Gejala klasik DM + Glukosa plasma sewaktu > 200 mg/dl
Gejala klasik DM + Glukosa plasma puasa > 126 mg/dl atau
Glukosa plasma 2 jam pada Tes Toleransi Glukosa Oral (TTGO) > 200 mg/dl, menggunakan beban glukosa 75gr *anhidrat* yang dilarutkan dalam air

C. Machine Learning

Istilah *Machine Learning* pertama kali disebutkan oleh Arthur Samuel pada tahun 1959, pada saat itu ia menjelaskan dalam konteks menyelesaikan permainan catur dengan mesin [15]. Secara istilah *machine learning* merupakan sebuah model komputasi statistik, yang berfokus pada prediksi menggunakan komputer. Algoritma *machine learning* membangun model matematika dari data sampel, yang dikenal sebagai "data pelatihan atau data *training*", untuk membuat prediksi atau keputusan tanpa diprogram secara eksplisit untuk melakukan tugas.



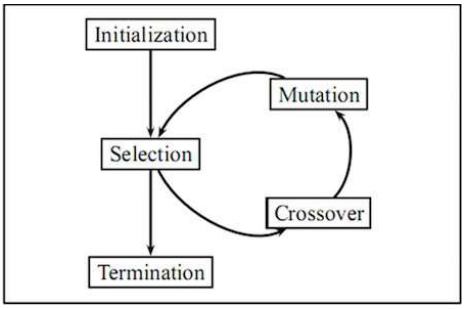
Gambar 3. Alur Kerja Machine Learning [10]

Inti alur kerja dari *machine learning* adalah tentang mengambil data mentah sebagai *input* dan menghasilkan prediksi sebagai *output*. Alur kerja *machine learning* dapat tersebut dapat dilihat pada Gambar. 3. Kemudian, secara luas algoritma *machine learning* dapat diklasifikasikan menjadi tiga jenis, yaitu *supervised learning, unsupervised learning,* dan *reinforcement learning.*

D. Algoritma Evolusi

Algoritma Evolusi berkembang seiring dengan perkembangan teknologi informasi yang sangat pesat. Karena kemampuannya untuk menyelesaikan berbagai masalah kompleks, algoritma ini banyak digunakan dalam bidang fisika, biologi, ekonomi, sosiologi, dan lain-lain yang sering menghadapi masalah optimasi. Gambar 4 memperlihatkan Algoritma Evolusi yang merupakan suatu algoritma pencarian berdasarkan mekanisme seleksi alam

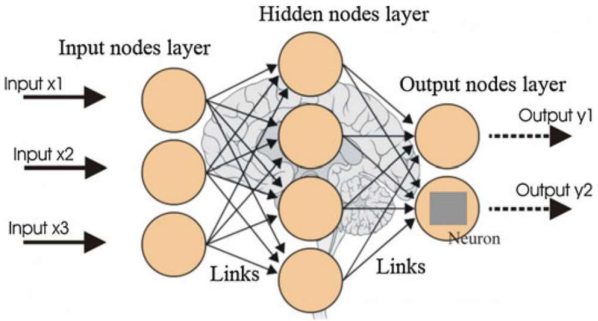
dan genetika alam. Algoritma evolusi dimulai dengan sekumpulan solusi awal (individu) yang disebut populasi. Satu hal yang sangat penting adalah bahwa satu individu menyatakan satu solusi. Populasi awal akan berevolusi menjadi populasi baru melalui serangkaian iterasi (generasi). Pada akhir iterasi, algoritma evolusi mengembalikan satu anggota populasi yang terbaik sebagai solusi untuk masalah yang dihadapi.



Gambar 4. Tahapan pada Algoritma Evolusi [16]

E. Artificial Neural Network

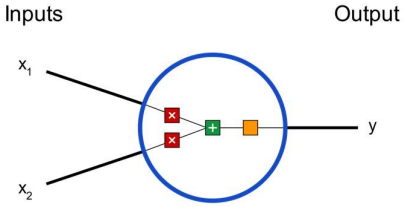
Menurut Fausett, *Artificial Neural Network* adalah suatu sistem pemrosesan informasi yang memiliki karakteristik-karakteristik menyerupai jaringan syaraf biologi [17]. Sebuah jaringan dalam *artificial neural network* merupakan kombinasi dari beberapa neuron. Jaringan tersebut terdiri dari sebuah lapisan *input* (*input layer*), sebuah lapisan *output* (*output layer*) dan kemungkinan satu atau lebih lapisan atau sering disebut sebagai lapisan tersembunyi (*hidden layer*). Setiap *layer* terdiri dari beberapa neuron dan neuron-neuron ini terhubung dengan neuron-neuron lain pada *layer* terdekat (Gambar 5).



Gambar 5. Arsitektur Artificial Neural Network [6].

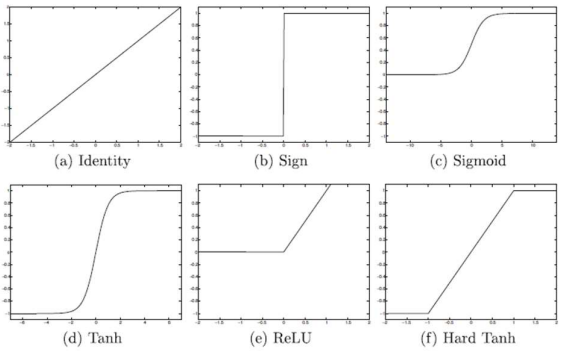
Secara umum proses pemodelan *artificial neural network* terbagi menjadi dua bagian, yaitu proses *training* dan *testing*. Proses *training* merupakan proses pembelajaran dari algoritma *artificial neural network* yang mengatur *input-input* serta bagaimana pemetaannya pada *output* sampai diperoleh model yang sesuai. Proses *training* terjadi pada saat pengaturan *weight* (bobot) dan bias. Sedangkan proses *testing* adalah proses pengujian ketelitian dari model yang telah diperoleh dari proses *training*. Dalam kajian secara statistik, proses *training* dan *testing* dapat disetarakan dengan proses estimasi dan *cross validation*.

Proses *learning* dimulai dari *neuron,* sebuah unit dasar dari *neural network.* Proses yang terjadi pada setiap *neuron* adalah menerima *input*, melakukan operasi-operasi matematis, dan menghasilkan *output*. Apabila terdapat *input* x_1 dan x_2, operasi pertama yang terjadi pada operasi [x] adalah mengalikan setiap *input* dengan nilai *weight* atau bobot dimana nilai *weight* dan *bias* akan diinisiasi secara acak. Setelah operasi perkalian *input* dengan *weight,* operasi dilanjutkan dengan penambahan nilai *bias* (Gambar 6).



Gambar 6. Ilustrasi sebuah neuron dalam artificial neural network [18].

Terakhir, hasil dari penjumlahan tersebut dimasukkan ke dalam fungsi aktivasi tertentu. Terdapat berbagai macam fungsi aktivasi, Aggarwal dalam bukunya menjelaskan bahwa terdapat 6 macam fungsi aktivasi, yaitu *Identity* (linier), *Sign, Sigmoid, Tanh, Rectified Linear Unit (ReLU),* dan *Hard Tanh.*



Dalam bentuk matematika, fungsi-fungsi aktivasi tersebut dapat dituliskan pada Tabel II berikut.

TABEL II. MACAM-MACAM FUNGSI AKTIVASI

Fungsi Aktivasi	Rumus
Identity atau linier	$f(x) = x$
Sign	$f(x) := \begin{cases} -1 & if\ x < 0 \\ 0 & if\ x = 0 \\ 1 & if\ x > 0 \end{cases}$
Sigmoid	$f(x) = \frac{1}{1+e^{-x}}$
Tanh	$f(x) = \frac{e^{2x}-1}{e^{2x}+1}$ atau $f(x) = 2 \cdot sigmoid(2x) - 1$
ReLU	$f(x) = \max\{x, 0\}$
Hard Tanh	$f(x) = \max\{\min[x, 1], -1\}$

Fungsi aktivasi berperan penting dalam *neural network* karena digunakan untuk mengubah suatu nilai yang tak terbatas menjadi suatu *output* dengan bentuk tertentu yang dapat digunakan sebagai nilai untuk prediksi. Saat ini *ReLU* menjadi pilihan fungsi aktivasi paling populer untuk digunakan pada *neural network,* terutama digunakan pada *hidden layer* [10]. Seluruh proses hingga melewati fungsi

aktivasi ini disebut fase *forward* atau fase maju dari *neural network*.

Kemudian proses dilanjutkan dengan menghitung nilai *error* antara nilai *output* untuk prediksi dengan nilai yang sesungguhnya atau nilai target dengan menghitung nilai *loss.* Nilai *loss* dapat dihitung menggunakan *Mean Squared Error (MSE)*. Rumus ini secara tidak langsung adalah nilai rata-rata dari nilai *squared error* yang biasa digunakan untuk menghitung jarak. Sehingga *MSE* merupakan rata-rata jarak kesalahan antara nilai yang diprediksi dengan nilai sesungguhnya. Sehingga semakin kecil nilai ini maka semakin baik prediksi yang dihasilkan.

Proses berlanjut pada tahapan *backpropagation,* yaitu tahapan untuk mengubah nilai *weight* sehingga nilai *loss* bisa menjadi lebih kecil lagi. Hal ini bisa dilakukan dengan menghitung nilai dari *loss gradient* menggunakan rumus turunan parsial. Nilai *loss gradient* tersebut kemudian dapat digunakan untuk mengubah nilai *weight* dengan menggunakan *Stochastic Gradient Descent (SGD)* sehingga dapat menurunkan nilai *loss* dari *neural network.* Dengan SGD, pengubahan dari nilai *weight* dapat dikontrol dengan adanya variabel *learning rate* (η).

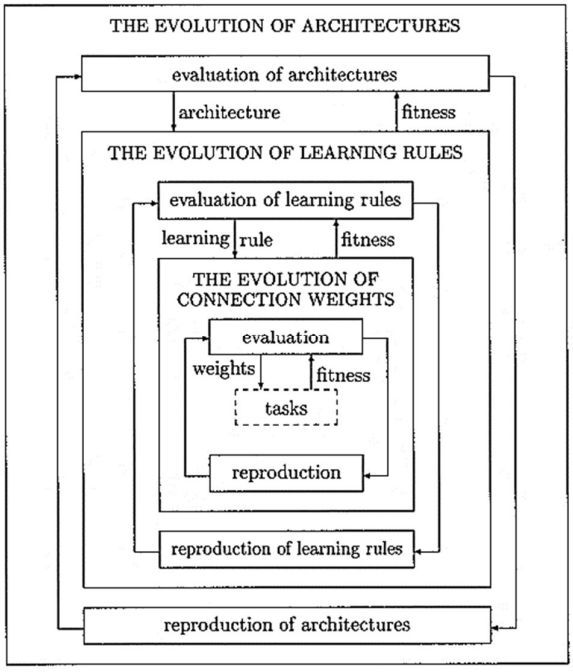
Apabila nilai dari *loss gradient* positif maka nilai *weight* akan berkurang, sebaliknya jika nilai dari *loss gradient* negatif maka nilai *weight* akan bertambah. Jika semua proses ini dilakukan terhadap semua nilai *weight* dan bias, maka secara perlahan-lahan proses ini akan menurunkan nilai *loss*. Proses yang dilakukan dari menghitung *error* hingga mengubah nilai dari *weight* serta *bias (backpropagation)* ini disebut fase *backward* atau fase mundur.

Dalam melakukan *training,* terdapat suatu istilah yang dikenal dengan *epoch.* Istilah tersebut menggambarkan satu siklus ketika *neural network* telah menjalankan satu kali fase maju dan fase mundur pada seluruh *data training. Epoch* merupakan suatu parameter yang dibutuhkan ketika menjalankan *training* pada *neural network.* Apabila *epoch* bernilai 2000, maka *neural network* akan melakukan 2000 siklus fase maju dan mundur terhadap *data training.* Pada saat proses *training* mencapai *epoch* ke 2000, maka proses *training* akan berhenti.

F. Evolving Artificial Neural Network

Evolving artificial neural network mengacu pada suatu kelas khusus dari *artificial neural network,* dimana evolusi menjadi suatu bentuk dasar lain untuk beradaptasi sebagai bagian dari tahapan *learning.* Algoritma evolusi digunakan untuk berbagai tugas seperti *training* pada *weight,* desain arsitektur, dan inisiasi nilai *weight* [9]. Satu fitur berbeda dari EANN adalah kemampuannya untuk beradaptasi dengan lingkungan. Dalam arti yang lebih luas, EANN dapat dianggap sebagai kerangka kerja umum untuk sistem adaptif, yaitu sistem yang dapat mengubah arsitektur dan aturan pembelajarannya dengan tepat tanpa campur tangan manusia.

Algoritma evolusi pada ANN dapat dilakukan pada tiga titik, yaitu: nilai *weight*, arsitektur, dan *learning rules*. Evolusi pada arsitektur memungkinkan ANN untuk menyesuaikan topologi mereka tanpa campur tangan manusia. Dengan demikian, hal ini memberikan pendekatan untuk menghasilkan desain arsitektur ANN secara otomatis karena bobot dan struktur koneksi ANN dapat dikembangkan. Gambar 7 memperlihatkan Evolusi *learning rules* dapat dianggap sebagai proses "*learning to learn*" pada ANN, di mana adaptasi *learning rules* dicapai melalui evolusi. Ini juga dapat dianggap sebagai *adaptive process of automatic discovery of novel learning rule* [9].



Gambar 7. Kerangka kerja umum algoritma EANN [9].

Siklus evolusi pada algoritma ini dimulai dengan membuat populasi berisi individu atau juga disebut gen dari model ANN yang memiliki nilai *weight*, data arsitektur dan *learning rule*. Proses pembuatan dilakukan secara acak *(randomly generate)* kemudian *train* setiap gen yang sudah dibuat. Setelah itu hitung nilai *fitness* dari setiap gen menggunakan nilai akurasi yang dicapai. Pilih *parents* dari seluruh populasi berdasarkan nilai *fitness* terbaik. Dari *parents* yang tercipta lakukan mutasi untuk menciptakan *offspring* atau *child* yang akan membentuk generasi baru.

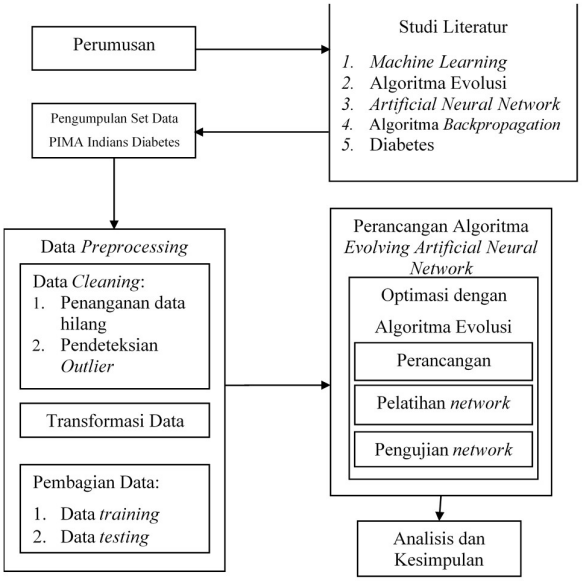
Prosedur pencarian global seperti algoritma evolusi biasanya mahal secara komputasi. Akan lebih baik untuk tidak melakukan evolusi pada seluruh aspek evolusi yang dapat dilakukan ada ANN. Namun, pencarian pada seluruh aspek sesungguhnya bermanfaat, terutama ketika tidak tersedianya *knowledge* yang dimiliki sebelumnya mengenai kinerja ANN pada suatu penelitian, karena pengujian secara *trial and error* dan metode heuristik lainnya akan kurang efektif dan efisien dalam keadaan seperti itu. Dengan seiring meningkatnya kemampuan *parallel computing*, evolusi pada ANN yang besar menjadi mungkin. Sehingga harapannya algoritma evolusi tersebut dapat menemukan kemungkinan arsitektur dan *learning rules* yang baru dan optimal.

III. METODE PENELITIAN

A. Desain Penelitian

Desain penelitian merupakan kerangka kerja yang akan digunakan untuk melakukan penelitian. Pada bagian ini

penulis akan memaparkan kerangka kerja mulai dari awal penelitian hingga selesai. Desain Penelitian dapat dilihat pada Gambar 8.



Gambar 8. Desain Penelitian

B. Metode Penelitian

Adapun metode yang dilakukan dalam penelitian ini dibagi ke dalam beberapa bagian, yaitu metode pengumpulan data, data *preprocessing,* dan perancangan algoritma *evolving artificial neural network*

1) Metode Pengumpulan data

Metode pengumpulan data yang akan digunakan pada penelitian ini didapatkan dari sumber PIMA Indians Diabetes data set. Data yang digunakan dapat diunduh pada sumber yang telah disebutkan pada poin bahan penelitian

2) Metode *Data Preprocessing*

Tahap data preprocessing merupakan tahapan setelah dilakukan pemeriksaan terhadap data yang diperoleh. Hal ini dilakukan dalam rangka melihat tingkat kualitas dari data yang diperoleh. Permasalahan yang ada pada data antara lain kemungkinan data yang diperoleh terlalu banyak, terlalu sedikit, data yang hilang, data aktual di luar hasil yang diinginkan (outlier), dan data yang diperoleh dalam skala yang berbeda. Hal ini dilakukan agar hasil prediksi yang diinginkan memiliki tingkat akurasi dan presisi yang tinggi. Tahapan ini meliputi tahapan pemeriksaan data dan proses pengolahan data.

Permasalahan yang ada setelah dilakukan pemeriksaan terhadap data dapat diselesaikan dengan beberapa solusi. Data yang terlalu banyak dapat diselesaikan dengan melakukan sampling terhadap data yang diperoleh. Data yang terlalu sedikit dan data yang hilang dapat diselesaikan dengan mengumpulkan kembali data yang dianggap masih kurang dan hilang tersebut. Data aktual yang bersifat outlier dapat diselesaikan dengan mereduksi data tersebut. Data dengan skala yang berbeda dapat diselesaikan dengan normalisasi data.

3) Perancangan Algoritma EANN

Penentuan model dari artificial neural network yang akan dipakai harus ditentukan dengan model manakah yang paling mudah beradaptasi dan yang paling sering dipakai untuk membandingkan setiap kemungkinan. Parameter yang terdapat untuk menjalankan algoritma evolving artificial neural network dalam melakukan prediksi adalah sebagai berikut:

a) Jumlah neuron *input*
Jumlah ini mudah ditentukan karena tergantung kepada banyaknya feature dari set data.

b) Jumlah neuron pada hidden layer
Teknik yang umum digunakan dalam menentukan jumlah neuron pada hidden layer adalah dengan percobaan. Hal penting yang harus diperhatikan dalah selalu memilih jaringan dengan performa terbaik dan jumlah hidden neuron yang sedikit.

c) Jumlah neuron output
Umumnya untuk melakukan prediksi dengan hanya terdapat dua kelas (biner) memakai satu output neuron saja.

d) Fungsi aktivasi
Fungsi aktivasi digunakan untuk menentukan output dari neuron yang diproses. Kriteria penentu yang dipakai untuk menentukan fungsi aktivasi adalah kemampuannya untuk mempercepat fase pembelajaran dan meningkatkan keakuratan dari artificial neural network.

e) Nilai learning rate
Nilai ini mempengaruhi seberapa jauh perubahan yang akan terjadi pada optimasi nilai weight.

f) Banyaknya gen dalam populasi
Banyaknya gen pada populasi akan meningkatkan variasi model yang tercipta, namun akan membutuhkan waktu inisiasi yang lebih lama.

g) Banyaknya generasi yang akan dievolusi
Banyaknya evolusi memungkinkan algoritma untuk dapat menemukan gen yang lebih baik, namun semakin banyak evolusi maka akan sangat memakan waktu untuk diproses.

h) Nilai mutation rate
Nilai ini merupakan nilai untuk probabilitas terjadinya mutasi pada gen untuk proses regenerasi.

IV. HASIL DAN PEMBAHASAN

A. Pengumpulan Data

Data yang digunakan dalam penelitian ini dapat diunduh secara bebas melalui tautan berikut ini https://www.kaggle.com/uciml/pima-indians-diabetes-database/download (Tabel III). Namun untuk mengunduh silakan melakukan registrasi akun Kaggle terlebih dahulu.

Set data ini berisi 768 baris data dengan 8 kolom *independent features*, yaitu *Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, Body Mass Indeks (BMI) dan Diabetes Pedigree Function (DPF)*. Kemudian 1 kolom *dependent features* bernama *Outcome* yang merupakan label hasil diagnosis positif atau negatif dari pasien.

TABEL III. PRATINJAU DATA PIMA INDIAN DIABETES

Pregnancies	Glucose	Blood Pressure	Skin Thickness	Insulin	BMI	DPF	Age	Outcome
1	89	66	23	94	28	0	21	0
0	137	40	35	168	43	2	33	1
3	78	50	32	88	31	0	26	1
1	103	30	38	83	43	0	33	0

B. Data Pre-processing

Setelah data berhasil diunduh, tahapan selanjutnya masuk ke dalam tahap data *preprocessing.* Dalam tahap ini terdapat beberapa bagian tahapan, mulai dari *cleaning, transformation, dan splitting.*

Tahapan pertama dimulai dari *data cleaning.* Tahapan ini bertujuan untuk mencari apakah terdapat data yang hilang ataupun *outlier* di dalam data set. *Outlier* mewakili konsep penting dalam teori *machine learning*. Meskipun, maknanya jelas, dampaknya pada proses *learning* tidak sederhana. *Outlier* adalah sampel dalam data pelatihan yang tidak mewakili tren generik dalam data. Dari sudut pandang matematika, jarak *outlier* dari sampel lain dalam data biasanya besar. Jarak yang begitu jauh dapat membuat model *machine learning* jauh dari perilaku yang diinginkan. Dengan kata lain, satu set outlier kecil dapat mempengaruhi proses *learning* model yang merugikan dan dapat mengurangi metrik secara signifikan. Oleh karena itu, ini merupakan bagian penting dari model *machine learning* [15]. Pendeteksian *outlier* akan dilakukan satu per satu pada setiap kolom data yang ada. Setelah melalui tahapan *cleaning* jumlah baris data yang ada berkurang dari 768 baris data menjadi 756 baris data.

Kemudian tahapan *data preprocessing* berlanjut ke proses *data transformation.* Dalam tahap ini transformasi yang akan dilakukan terhadap data adalah melakukan standarisasi. Tujuannya adalah untuk mengubah nilai numerik dalam setiap variabel sehingga setiap variabel memiliki nilai rata-rata dan variansi bernilai 0. Efek positif dari standarisasi data adalah untuk menyusutkan besarnya nilai suatu variabel dan mengubahnya menjadi skala yang lebih proporsional.

Tahapan ini bahkan dijadikan sebagai prasyarat dalam beberapa algoritma *machine learning.* Dalam *neural network*, tahap ini penting untuk memastikan bahwa algoritma *backpropagation* berfungsi sebagaimana seharusnya. Seperti yang dapat dilihat sebelumnya, variabel seperti Insulin dan Diabetes Pedigreee Function memiliki skala yang sangat berbeda, nilai maksimum untuk Insulin adalah 846 sedangkan nilai maksimum untuk Fungsi Diabetes Pedigreee hanya 2,42. Dengan skala yang berbeda seperti itu, variabel dengan skala yang lebih besar cenderung mendominasi pada tahap *training neural network* [10].

TABEL IV. PRATINJAU DATA SETELAH DITRANSFORMASI

Pregnancies	Glucose	Blood Pressure	Skin Thickness	Insulin	BMI	DPF	Age
-0.8567380 8	-1.2077916	-0.5504921 5	0.2809209 1	-0.5638990 8	-0.8478921 2	-0.3654924 8	-0.1849528 6
-0.8567380 8	-1.0748812 1	0.5504921 5	0.3636757 7	-0.2667456	-0.6273846	-0.9299147 7	1.0385814 2
0.3618771 4	-0.1777360 5	0.1287734	-0.6508709 1	-0.5638990 8	-0.9948971 3	-0.8256193 4	-0.2703157 1
1.8851461 7	-0.2109636 4	-0.2874323 4	-0.6508709 1	0.5638990 8	0.4310515 1	-1.0311426 8	-0.3556785 7
0.0572233 3	-0.3771016 4	1.6571208 7	-0.6508709 1	0.5638990 8	0.7691630 4	-0.8562944 7	-0.2703157 1

Tahapan selanjutnya merupakan *data splitting,* pada tahap ini data akan dibagi menjadi empat, yaitu X_train, X_test, y_train dan y_test. Data dalam X_train merupakan variabel independen atau *features* yang akan digunakan pada tahapan *training*, sedangkan X_test akan digunakan pada tahap *testing.* Kemudian untuk data dalam y_train merupakan variabel dependen atau target yang akan digunakan pada tahapan *training,* sedangkan y_test akan digunakan pada tahap *testing.*

Tujuan membagi data ke dalam set *training* dan *testing* adalah untuk menghindari *overfitting* dan untuk menyediakan sumber data yang tidak bias untuk mengevaluasi kinerja model. Biasanya, set *training* digunakan untuk menyesuaikan dan meningkatkan performa model. Kemudian, set *testing* digunakan untuk penghentian proses pelatihan, yaitu ketika kinerja model pada set *testing* berhenti membaik. Ini semua digunakan untuk menghindari *overfitting* pada *neural network* [10]. Set *testing* juga dikenal sebagai set *holdout*, karena *neural network* tidak akan pernah dilatih menggunakan set ini. Sebagai gantinya, set *testing* digunakan untuk mengevaluasi model di akhir. Hasil evaluasi ini memberi refleksi akurat dari kinerja model dunia nyata.

Dalam praktik para peneliti umumnya membagi data menjadi 2 bagian dengan perbandingan 80% *training* dan 20% *testing* [10]. Kemudian satu hal penting yang perlu diperhatikan adalah bahwa proses data *splitting* harus dilakukan secara acak. Jika menggunakan metode non-acak untuk membagi data (misalnya, 80% baris awal menjadi *training* dan 20% baris terakhir menjadi *testing*), hal ini berpotensi dapat memasukkan bias ke dalam *set training* dan set training. Sebagai contoh, terdapat data yang dapat diurutkan dalam urutan kronologis, kemudian apabila menggunakan metode pembagian data yang non-acak hal ini berarti bahwa model hanya dilatih tentang data dari tanggal tertentu, yang sangat bias dan tidak akan bekerja dengan baik

di dunia nyata. Jadi setelah proses pembagian data, terdapat 604 baris serta 8 kolom untuk set X_train, 152 baris serta 8 kolom untuk set X_test, 604 baris serta 1 kolom untuk set y_train, dan 152 baris serta 1 kolom untuk set y_test.

C. Rancangan Skenario Eksperimen

Eksperimen dilakukan dengan beberapa skenario, yaitu dengan melakukan proses *machine learning* terhadap set data asli sebelum diproses pada tahap data *preprocessing* dan set data yang telah melalui tahap *preprocessing.* Hal ini dilakukan untuk dapat menilai seberapa efektif proses dari *data preprocessing* yang telah dilakukan. Kemudian skenario tersebut disertai juga dengan melakukan beberapa kali pelatihan dengan parameter algoritma evolusi yang berbeda. Sehingga, ketika telah ditemukan eksperimen dengan hasil terbaik menggunakan data yang telah melalui tahap *preprocessing,* maka dilakukanlah eksperimen menggunakan data asli dengan menggunakan parameter dari hasil tersebut. Tabel V berikut untuk memperjelas berbagai skenario yang dilakukan dalam pengolahan data.

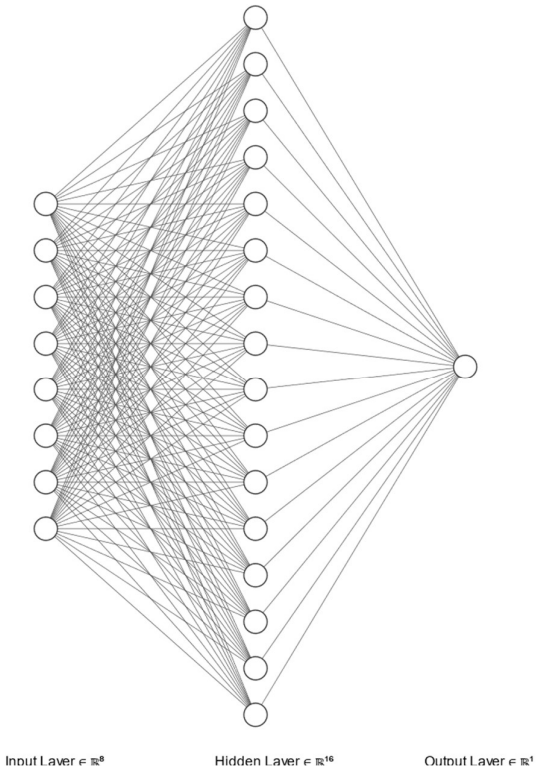
TABEL V. SKENARIO PARAMETER PADA EKSPERIMEN EANN

No	Parameter		
	nPopulation	***nGeneration***	***mutationRate***
1	10	100	0,25
2	10	100	0,5
3	10	100	0,75
4	10	500	0,25
5	10	500	0,5
6	10	500	0,75
7	25	100	0,25
8	25	100	0,5
9	25	100	0,75
10	25	500	0,25
11	25	500	0,5
12	25	500	0,75
13	25	1000	0,25
14	25	1000	0,5
15	50	100	0,25
16	50	100	0,5
17	50	100	0,75
18	50	500	0,25
19	50	500	0,5
20	50	500	0,75
21	50	1000	0,25
22	50	1000	0,5
23	100	100	0,25
24	100	100	0,5
25	100	100	0,75
26	100	500	0,25
27	100	500	0,5
28	100	500	0,75
29	100	1000	0,25
30	100	1000	0,5

Dalam penelitian ini penulis juga akan menjalankan algoritma sejenis dengan menggunakan *software library* sebagai pembanding algoritma yang telah dibangun. Algoritma yang akan digunakan adalah *Artificial Neural Network* menggunakan *libary* scikit-learn dan juga *library* keras. Adapun berbagai parameter yang akan digunakan akan mengikuti parameter dari hasil eksperimen terbaik yang didapatkan pada pelatihan algoritma *Evolving Artificial Neural Network*.

D. Hasil

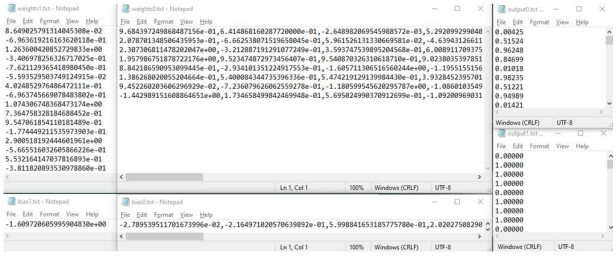
Setelah melakukan *training* sesuai dengan rancangan skenario eksperimen yang telah dibuat, penulis melakukan pencatatan terhadap performa dari hasil seluruh proses *training.* Performa yang dimaksud meliputi waktu eksekusi, akurasi, *confusion matrix, precision, recall,* dan *f1-score.* Menurut data, hasil terbaik mendapatkan akurasi sebesar 83,55%. Hasil tersebut disertai juga dengan parameter untuk model ANN dengan banyaknya neuron pada *hidden layer* sebanyak 16 neuron, *learning rate* bernilai 0,02, dan menggunakan fungsi aktivasi *Tanh*. Hasil tersebut didapat dari beberapa skenario eksperimen, yaitu skenario nomor 2, 5, 6, 8, 11, 14, 15, 16, 18, 19, 20, 21, 22, 27, 28, dan 30.



Gambar 9. Model arsitektur dari hasil dengan akurasi terbaik

Output dari algoritma *evolving artificial neural network* yang dibahas merupakan algoritma yang dibangun berdasarkan pengetahuan penulis yang bersumber dari jurnal, buku, dan beberapa sumber *online*. Berdasarkan hasil eksperimen, ditemukan bahwa terdapat beberapa data yang memiliki *output* yang sama. Jika dirangkum, hanya terdapat enam model yang dihasilkan dari seluruh proses *training*. Rangkuman tersebut dapat dilihat pada Tabel VII.

Kemudian untuk menyimpan hasil terbaik dari proses *training,* nilai *weight* dan *bias* dari setiap layer model serta *output* prediksi disimpan ke dalam beberapa file berformat *.txt.



Gambar 10. *Output* model terbaik dari proses pelatihan

Menurut Tabel VII, parameter yang digunakan dan menghasilkan *output* tersebut adalah sebagai berikut.

TABEL VI. PARAMETER YANG DIGUNAKAN PADA RANGKUMAN *OUTPUT* DARI ALGORITMA *EANN*

No.	Parameter		
	nPopulation	***nGeneration***	***mutationRate***
1	100	100	0,25
2	10	100	0,25
3	100	100	0,5
4	25	100	0,25
5	25	500	0,75
6	10	100	0,5

Kemudian, dilakukanlah *training* dengan menggunakan parameter yang terdapat pada Tabel VI terhadap set data asli yang belum melalui tahap *preprocessing.* Hasil dari proses *training* tersebut dapat dilihat pada Tabel VIII. Berdasarkan tabel tersebut, skenario ini menghasilkan akurasi terbaik sebesar 80,52%. Model yang didapat memiliki 8 neuron, *learning rate* bernilai 0,14, dan dengan menggunakan fungsi aktivasi *Tanh.* Dengan demikian, hal ini menandakan bahwa efektivitas dari tahapan *data preprocessing* dapat meningkatkan akurasi hingga sebesar 3,03%.

Sebagaimana telah dijelaskan sebelumnya, pada penelitian ini akan dilakukan perbandingan antara program *native* yang dibuat oleh penulis dan program dengan menggunakan *software library.* Dalam hal ini, *sofware library* yang akan digunakan adalah *Scikit-learn* dan *Keras.* Perbandingan akan dilakukan dengan menggunakan parameter dari hasil yang dihasilkan dari algoritma *evolving artificial neural network,* yaitu sesuai dengan tabel parameter rangkuman *output* pada Tabel VI.

Pengujian algoritma pembanding pertama dijalankan menggunakan *software library* Scikit-learn. Berdasarkan hasil pengujian, akurasi terbesar yang didapatkan bernilai 80%. Sedangkan pada pengujian kedua algoritma pembanding yang digunakan adalah *software library* Keras. Berdasarkan hasil pengujian, akurasi terbesar yang didapatkan bernilai 80,921% hampir bernilai 81%.

Korespondensi: *yudi.a.hambali@upi.edu*

TABEL VII. RANGKUMAN *OUTPUT* DARI PROSES TRAINING ALGORITMA *EVOLVING ARTIFICIAL NEURAL NETWORK*

No	Parameter			Waktu Eksekusi	Neuron	*Learning Rate*	Fungsi Aktivasi	Akurasi	*Confusion Matrix*				*Precision*		*Recall*		*F1*	
	nPopulation	*nGeneration*	*mutation Rate*						*TP*	*FP*	*FN*	*TN*	*0*	*1*	*0*	*1*	*0*	*1*
1	100	100	0,25	0:07:25	8	0,14	Linear	0,79605263	92	13	18	29	0,84	0,69	0,88	0,62	0,86	0,65
2	10	100	0,25	0:10:15	128	0,02	Sigmoid	0,80263157	92	13	17	30	0,84	0,7	0,88	0,64	0,86	0,67
3	100	100	0,5	0:11:52	8	0,15	Linear	0,80263157	89	16	17	30	0,84	0,65	0,85	0,64	0,84	0,65
4	25	100	0,25	0:07:19	32	0,04	Tanh	0,80921052	91	14	15	32	0,86	0,7	0,87	0,68	0,86	0,69
5	25	500	0,75	0:49:55	16	0,01	Tanh	0,81578947	89	16	12	35	0,88	0,69	0,85	0,74	0,86	0,71
6	10	100	0,5	0:04:23	16	0,02	Tanh	0,83552631	89	16	9	38	0,91	0,7	0,85	0,81	0,88	0,75

TABEL VIII. *OUTPUT* DARI PROSES *TRAINING* DENGAN ALGORITMA *EANN* MENGGUNAKAN SET DATA ASLI.

No	Parameter			Waktu Eksekusi	Neuron	*Learning Rate*	Fungsi Aktivasi	Akurasi	*Confusion Matrix*				*Precision*		*Recall*		*F1*	
	nPopulation	*nGeneration*	*mutation Rate*						*TP*	*FP*	*FN*	*TN*	*0*	*1*	*0*	*1*	*0*	*1*
1	100	100	0,25	0:04:51	8	0,14	Tanh	0,80519480	94	11	20	29	0,82	0,72	0,9	0,59	0,86	0,65
2	10	100	0,25	0:01:52	8	0,14	Tanh	0,80519480	94	11	19	30	0,83	0,73	0,9	0,61	0,86	0,67
3	100	100	0,5	0:07:20	8	0,14	Tanh	0,80519480	94	11	19	30	0,83	0,73	0,9	0,61	0,86	0,67
4	25	100	0,25	0:02:09	8	0,11	Tanh	0,80519480	94	11	19	30	0,83	0,73	0,9	0,61	0,86	0,67
5	25	500	0,75	0:42:40	8	0,13	Tanh	0,80519480	94	11	19	30	0,83	0,73	0,9	0,61	0,86	0,67
6	10	100	0,5	0:11:27	16	0,2	Sigmoid	0,79870129	97	8	23	26	0,81	0,76	0,92	0,53	0,86	0,63

TABEL IX. OUTPUT ALGORITMA ARTIFICIAL NEURAL NETWORK DENGAN LIBRARY SCIKIT-LEARN

No	*Neuron*	*Learning Rate*	Fungsi Aktivasi	Akurasi	*Confusion Matrix*			
					TP	*FP*	*FN*	*TN*
1	8	0,14	Linear	0,77	86	19	16	31
2	128	0,02	Sigmoid	0,8	93	12	19	28
3	8	0,15	Linear	0,78	86	19	15	32
4	32	0,04	Tanh	0,8	91	14	17	30
5	16	0,01	Tanh	0,74	86	19	21	26
6	16	0,02	Tanh	0,8	90	15	15	32

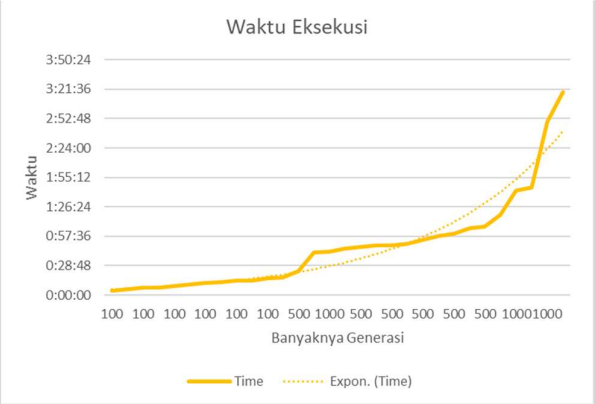
TABEL X. OUTPUT ALGORITMA ARTIFICIAL NEURAL NETWORK DENGAN LIBRARY KERAS

No	*Neuron*	*Learning Rate*	Fungsi Aktivasi	Akurasi	*Confusion Matrix*			
					TP	*FP*	*FN*	*TN*
1	8	0,14	Linear	0,809	93	12	17	30
2	128	0,02	Sigmoid	0,730	100	5	36	11
3	8	0,15	Linear	0,809	93	12	17	30
4	32	0,04	Tanh	0,802	91	14	16	31
5	16	0,01	Tanh	0,802	92	13	17	30
6	16	0,02	Tanh	0,802	92	13	17	30

E. Pembahasan

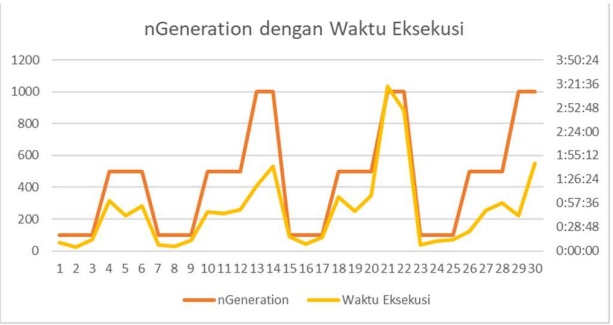
Berdasarkan paparan dari berbagai eksperimen yang telah dijelaskan pada subbab sebelumnya, penulis dapat melakukan beberapa pembahasan yang akan dipaparkan pada subbab ini. Menurut hasil dari algoritma *evolving artificial neural network,* nilai akurasi paling rendah yang didapatkan bernilai 79,60% dan akurasi paling tinggi bernilai 83,55% dengan rata-rata akurasi berada pada angka 82,17%. Artinya kemampuan algoritma untuk melakukan pelatihan cukup stabil karena rentang jarak antara akurasi terendah dan tertinggi tidak terlalu jauh.

Namun, jika dilihat pada sisi waktu eksekusi terdapat perbandingan yang cukup signifikan antara waktu eksekusi tercepat dan juga waktu eksekusi terlama. Waktu tercepat dilakukan dalam waktu 4 menit 23 detik dengan melakukan evolusi sebanyak 100 kali pada populasi berisi 10 gen. Sedangkan waktu terlambat tercatat selama 3 jam 18 menit 45 detik dengan melakukan evolusi sebanyak 1000 kali pada populasi berisi 50 gen. Bahkan jika diurutkan dari waktu eksekusi tercepat, peningkatan waktu eksekusi terhadap banyaknya generasi cukup bersifat eksponensial seperti pada Gambar 11.



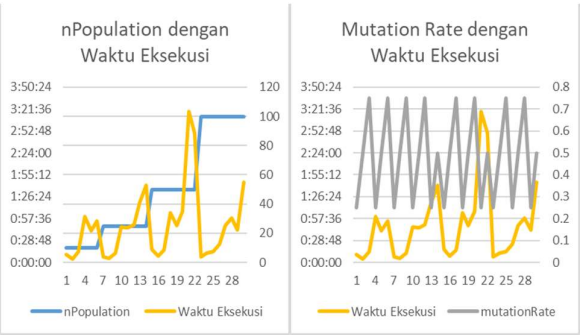
Gambar 11. Peningkatan waktu eksekusi pada algoritma *EANN*

Kemudian, melakukan perbandingan antara waktu eksekusi dengan parameter yang digunakan untuk proses *training*, yaitu *nPopulation, nGeneration,* dan *mutationRate.*



Gambar 12. Perbandingan *nGeneration* dengan waktu eksekusi

Pada Gambar 12 bahwa terdapat hubungan antara banyaknya *nGeneration* dengan waktu eksekusi. Waktu eksekusi cenderung ikut naik ketika banyaknya generasi meningkat. Begitu juga ketika banyaknya generasi turun, waktu eksekusi juga cenderung menurun.



Gambar 13. Perbandingan *nPopulation* dan *mutationRate* dengan waktu eksekusi

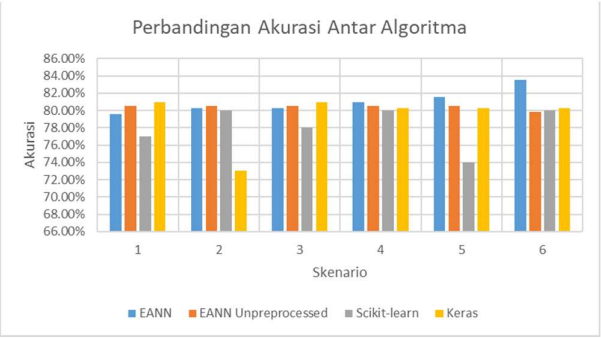
Sedangkan, jika dilihat pada Gambar 13, kedua parameter tersebut terlihat tidak terlalu memiliki kecenderungan atau hubungan satu sama lain. Jadi parameter utama yang mempengaruhi waktu eksekusi adalah banyaknya generasi pada proses *training*.

Lalu, berikut ini merupakan perbandingan akurasi dan rata-rata akurasi dari seluruh algoritma yang diuji yaitu *evolving artificial neural network* dengan menggunakan set data yang melalui tahap *preprocessing* dan set data asli tanpa melalui tahap *preprocessing* serta *artificial neural network* dengan menggunakan *software library* Scikit-learn dan Keras. Nilai akurasi tertinggi didapatkan melalui *training* dengan menggunakan algoritma *evolving artifical neural network* yang dikembangkan oleh penulis. Begitu juga jika dilihat dari sisi rata-rata akurasi yang didapatkan

TABEL XI. PERBANDINGAN AKURASI PADA SELURUH ALGORITMA

No.	EANN	EANN *Unpreprocessed*	Scikit-learn	Keras
1	79,61%	80,52%	77,00%	80,92%
2	80,26%	80,52%	80,00%	73,03%
3	80,26%	80,52%	78,00%	80,92%
4	80,92%	80,52%	80,00%	80,26%
5	81,58%	80,52%	74,00%	80,26%
6	83,55%	79,87%	80,00%	80,26%
Rata-rata	81,03%	80,41%	78,17%	79,28%

Perbedaan akurasi antara seluruh algoritma di setiap skenario eksperimen tidak terlampau jauh satu sama lain.



Gambar 14. Perbandingan akurasi antar algoritma

V. KESIMPULAN

Berdasarkan serangkaian proses penelitian yang sudah dilaksanakan, dapat ditarik beberapa kesimpulan sebagai berikut.

a) Terdapat beberapa analisis dan penyesuaian pada set data yang digunakan pada tahapan *preprocessing* khususnya tahap *data cleaning.* Berdasarkan pengujian yang sudah dilakukan, hal tersebut efektif dapat meningkatkan akurasi dari algoritma *evolving artificial neural network* yang digunakan

b) Dalam proses pengembangan seluruh program, bahasa Python sangat membantu kelancaran penelitian. Karena banyak sekali software library yang dimiliki oleh bahasa pemrograman ini, mulai dari proses data preprocessing, pengolahan data, hingga pembanding algoritma.

c) Penggunaan algoritma evolving artificial neural network terbukti efektif dalam menemukan model terbaik karena model yang dihasilkan memiliki akurasi yang paling tinggi jika dibandingkan dengan algoritma pembanding yang digunakan. Namun proses pelatihan dengan algoritma ini memakan waktu yang semakin lama jika banyaknya generasi juga semakin banyak.

REFERENSI

[1] World Health Organization, "The Top 10 Causes of Death," World Health Organization (WHO), 2018. [Daring]. Tersedia pada: https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death.

[2] World Health Organization, "Diabetes: Fakta dan Angka di Indonesia," 2016. [Daring]. Tersedia pada: http://www.searo.who.int/indonesia/topics/8-whd2016-diabetes-facts-and-numbers-indonesian.pdf.

[3] M. M. Roche dan P. P. Wang, "Factors associated with a diabetes diagnosis and late diabetes diagnosis for males and females," *J. Clin. Transl. Endocrinol.*, vol. 1, no. 3, hlm. 77–84, Sep 2014, doi: 10.1016/j.jcte.2014.07.002.

[4] International Diabetes Federation, *IDF Diabetes Atlas Ninth edition 2019*, 9 ed. International Diabetes Federation (IDF), 2019.

[5] International Diabetes Federation, *IDF Diabetes Atlas Eighth edition 2017*, 8 ed. International Diabetes Federation (IDF), 2017.

[6] S. Shanmuganathan dan S. Samarasinghe, Ed., *Artificial Neural Network Modelling*, vol. 628. Cham: Springer International Publishing, 2016.

[7] J. V. Tu, "Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes," *J. Clin. Epidemiol.*, vol. 49, no. 11, hlm. 1225–1231, Nov 1996, doi: 10.1016/S0895-4356(96)00002-9.

[8] S. Pratomo dan R. N. Dayawati, "Implementasi Metode Time Series Menggunakan Algoritma Evolving Artificial Neural Network Pada Kasus Peramalan Saham Implementation Of Time Series Method Using Evolving Artificial Neural Network In Stock Forecasting," hlm. 6, 2009.

[9] X. Yao, "Evolving Artificial Neural Networks," *Proc. IEEE*, vol. 87, no. 9, hlm. 25, 1999.

[10] J. Loy, *Neural network projects with Python: the ultimate guide to using Python to explore the true power of neural networks through six projects*. 2019.

[11] I. P. D. Lesmana, "Pengembangan Decision Tree J48 Untuk Diagnosis Penyakit Diabetes Mellitus," hlm. 5, 2012.

[12] A. Setiadi, "Penerapan Algoritma Multilayer Perceptron Untuk Deteksi Dini Penyakit Diabetes," no. 1, hlm. 14, 2012.

[13] R. N. Fatimah, "Diabetes Melitus Tipe 2," *J Major. Med. Fac. Lampung Univ.*, vol. 4, no. 5, hlm. 9, Feb 2015.

[14] P. E. I. PERKENI, "Konsensus Pengelolaan dan Pencegahan Diabetes Melitus Tipe 2 Di Indonesia." PB. PERKENI, 2015, [Daring]. Tersedia pada: https://pbperkeni.or.id/wp-content/uploads/2019/01/4.-Konsensus-Pengelolaan-dan-Pencegahan-Diabetes-melitus-tipe-2-di-Indonesia-PERKENI-2015.pdf.

[15] A. V. Joshi, *Machine Learning and Artificial Intelligence*. Cham: Springer International Publishing, 2020.

[16] B. Xu, H. Lin, K. B. Wagholikar, Z. Yang, dan H. Liu, "Identifying protein complexes with fuzzy machine learning model," *Proteome Sci.*, vol. 11, no. Suppl 1, hlm. S21, 2013, doi: 10.1186/1477-5956-11-S1-S21.

[17] L. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Prentice-Hall, 1994.

[18] V. Zhou, "Machine Learning for Beginners: An Introduction to Neural Networks," Jul 24, 2019. https://victorzhou.com/blog/intro-to-neural-networks/.