Pemanfaatan Pembelajaran Mendalam untuk *Domain Knowledge* : Klasifikasi pada Penyakit Jantung

Kevin Sean Hans Lopulalan[1], Mochamad Fadlan Adhari[2]

Program Studi Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam

Institut Teknologi Bandung

e-mail: kevinivec@gmail.com[1], fadlanadhari20@gmail.com [2]

Abstrak

Tak jarang manusia memiliki kelainan jantung karena satu dan lain hal. Untuk memeriksa keadaan penyakit jantung atau tidak, diagnosa bisa dilakukan pada gejala-gejala yang muncul pada tubuh. Deteksi gejala tersebut harus diolah agar menghasilkan prediksi yang akurat. Maka dari itu, salah satu cara yang dapat dilakukan adalah memanfaatkan data historis pasien penyakit jantung untuk memprediksi gejala tersebut menggunakan Pembelajaran Mendalam. Pembelajaran Mendalam merupakan salah satu metode dalam kecerdasan buatan dengan konsep yang terinspirasi dari otak manusia. Dengan Pembelajaran Mendalam ini, dapat dilihat suatu pola atau prediksi menggunakan data-data historis yang dimiliki. Artikel ini membahas terkait dengan perbandingan model-model Pembelajaran Mendalam untuk melakukan klasifikasi data pasien, terutama model MLP dan CNN. Pada MLP, digunakan tiga *package* yang berbeda untuk membangun modelnya dan pada CNN digunakan dua jenis teknik reduksi dimensi fitur. Variasi yang digunakan adalah *layer,* epochs, fitur, dan juga learning rate. Melalui beberapa simulasi tersebut dapat disimpulkan bahwa MLP cukup unggul dibandingkan dengan CNN karena data digunakan secara menyeluruh dan tidak ada informasi yang terbuang. Namun demikian, seluruh model yang dihasilkan memiliki akurasi yang cukup baik, yaitu lebih dari 70%.

Kata Kunci : Pembelajaran Mendalam, MLP, CNN, Klasifikasi

Abstract

Heart disorders are a common occurrence among individuals due to various factors. To diagnose heart disease, an assessment of the symptoms manifesting in the body is crucial. Accurate prediction of these symptoms can be achieved by processing the detected symptoms. One approach involves utilizing historical data of heart disease patients to predict these symptoms using Deep Learning (DL). DL is a subfield of artificial intelligence inspired by the human brain. It enables pattern recognition and prediction using historical data. This article delves into a comparative analysis of DL models for patient data classification, specifically focusing on MLP and CNN models. For MLP, three different packages are employed to construct the model, while for CNN, two feature dimension reduction techniques are utilized. The variations considered include layers, epochs, features, and learning rate. Through simulations, it is concluded that MLP outperforms CNN due to its comprehensive utilization of data, preventing information loss. Nevertheless, all the developed models exhibit remarkable accuracy, exceeding 70%.

Keywords: Deep Learning, MLP, CNN, Classification

1. Pendahuluan

Pembelajaran Mendalam merupakan salahsatu metode dalam kecerdasan buatan dengan konsep yang terinspirasi dari otak manusia. Pemodelan dengan Pembelajaran Mendalam ini dapat mengenali suatu pola kompleks untuk banyak aspek, seperti gambar, teks, suara, dan data lainnya yang dapat digunakan untuk melakukan prediksi cukup akurat. Salah satu keunggulan dari model Pembelajaran Mendalam ini adalah dapat melakukan klasifikasi berdasarkan data yang dimiliki.

Pada beberapa literatur, mulai banyak bermunculan metode yang dilakukan sebagai penerapan dari Pembelajaran Mendalam ini, misalnya *Multi Layer Perceptron* (MLP), *Recurrent Neural Network* (RNN), *Convolutional Neural Network* (CNN), dan juga *Long Short Term Memory* (LSTM). Model-model tersebut memiliki berbagai macam kegunaan. Dalam hal untuk klasifikasi, model Pembelajaran Mendalam yang cocok dan dapat digunakan adalah MLP dan CNN.

Banyak cara yang dapat dilakukan untuk memanfaatkan arsitektur yang dimiliki oleh CNN dan MLP, terutama di bidang kesehatan. Pada bidang kesehatan, model tersebut dapat digunakan untuk klasifikasi permasalahan penyakit pasien. Hal tersebut dapat terjadi karena banyaknya data historis pasien terdahulu yang dapat digunakan untuk melakukan prediksi/klasifikasi penyakit pasien dengan kondisi yang serupa. Maka dari itu, model ini sangat berguna terutama untuk melakukan diagnosa awal terkait penyakit yang dialami pasien sebelum ditindak lebih lanjut, terutama terkait penyakit jantung.

Jantung merupakan organ yang cukup penting bagi kehidupan manusia. Fungsi dari jantung adalah untuk memompa darah ke seluruh tubuh dan menampungnya kembali setelah organ paru-paru membersihkan darah tersebut. Namun dengan fungsi penting tersebut, tak jarang beberapa orang memiliki kelainan jantung karena beberapa hal. Hal-tersebut dapat terjadi karena pola hidup yang kurang baik. Beberapa gejala mulai muncul pada saat mengalami kelainan pada jantungnya. Untuk melakukan diagnosa, pihak dokter biasanya membutuhkan data-data kondisi pasien seperti gula darah, kolesterol, usia, jenis kelamin, dan lainnya. Hal ini dibutuhkan oleh pihak dokter untuk dapat menentukan keberadaan penyakit pada jantung pasien. Untuk itu, diperlukan suatu model yang dapat menerima data-data kondisi pasien untuk menentukan keberadaan penyakit jantung tersebut.

Pada artikel ini, akan dicoba untuk dilakukan klasifikasi penyakit jantung terkait dengan data yang diberikan. Data tersebut berisi terkait kondisi-kondisi pasien yang sudah diketahui keadaan jantungnya (normal atau tidak). Dari data tersebut, akan dicoba dimodelkan menggunakan CNN dan MLP terkait klasifikasinya dan dibandingkan masing-masing hasilnya.

2. Metodologi

Untuk keperluan analisis, digunakan data pasien penyakit jantung yang diambil dari https://kaggle.com dengan banyak data kurang lebih 650 pasien. Kondisi-kondisi yang diberikan meliputi :

- Umur
- Jenis kelamin (Pria atau Wanita)
- Keterangan sakit dada (terdapat 4 nilai, diduga *Typical Angina, Atypical Angina, Non-anginal Chest Pain, Asymptomatic*)
- Tekanan darah saat istirahat
- Kadar kolesterol serum dalam mg/dl
- Gula darah puasa
- Hasil elektrokardiografi saat istirahat
  Keterangan : 0 untuk normal, 1 untuk kelainan gelombang ST-T, dan 2 untuk hipertrofi ventrikel kiri atau kelainan lainnya
- Detak jantung maksimum
- Angina karena olahraga
- Oldpeak : depresi ST yang disebabkan oleh olahraga relatif terhadap istirahat
- Kemiringan segmen ST puncak latihan
- Jumlah pembuluh darah utama saat fluoroskopi (0-3)
- Thallium
  Keterangan : 0 untuk normal, 1 untuk *fixed defect*, dan 2 untuk *reversible defect*
- Target
  Keterangan : 0 untuk normal, 1 untuk adanya penyakit jantung

Dari data yang ada, akan dilakukan pembentukan model klasifikasi menggunakan Pembelajaran Mendalam. Model yang digunakan pada artikel ini adalah MLP menggunakan Numpy, MLP menggunakan TensorFlow, CNN dengan teknik reduksi PCA, dan juga CNN dengan teknik reduksi seleksi. Berikut ini adalah rincian arsitektur model yang digunakan

2.1 *Multi Layer Perceptron* (MLP)

Multi Layer Perceptron atau MLP adalah salah satu model atau arsitektur dari Pembelajaran Mendalam yang merupakan konstruksi modern dari *Feed Forward Neural Network* (FNN). Model ini memuat neuron yang saling terhubung penuh dilengkapi dengan fungsi aktivasi nonlinear. Terdapat tiga jenis MLP yang akan dibangun pada artikel ini yaitu menggunakan numpy, TensorFlow, dan Sklearn di Python.

Ketiganya memiliki arsitektur yang sama, yaitu adanya *input layer*, *hidden layer,* dan *output layer*. Arsitektur untuk masing-masing metoda akan dijelaskan lebih terperinci pada bagian selanjutnya.

2.2 *Convolutional Neural Network* (CNN)

Convolutional Neural Network atau CNN adalah salah satu model dari Pembelajaran Mendalam yang dapat digunakan untuk melakukan *image recognition, image classifications*, dan lain-lain yang berhubungan dengan data spasial. Pada artikel ini, data yang disiapkan adalah suatu data berbentuk numerikal yang memiliki 13 variabel berbeda. Untuk keperluan CNN, data-data tersebut harus diubah menjadi suatu gambar (pixel) dan dikehendaki berbentuk persegi. Maka dari itu, data tersebut akan direduksi menjadi sebanyak kuadrat sempurna (misalnya 4 atau 9). Terdapat dua cara reduksi yang dilakukan, yaitu :

2.2.1. *Principal Component Analysis* (PCA)

PCA adalah suatu teknik untuk mereduksi dimensi namun tetap mempertahankan informasi yang didapatkan dari data aslinya. Berikut ini adalah langkah-langkah untuk menggunakan teknik PCA :

1. Standarisasi data : mengurangi seluruh data dengan rata-rata dan melakukan penskalaan ke variansi satuan
2. Menghitung matriks kovariansi : matriks kovariansi merangkum seluruh pasangan antar fitur pada data yang dihitung berdasarkan data yang sudah distandarisasi
3. Menghitung nilai eigen dan vektor eigen : nilai eigen dan vektor eigen dari matriks kovariansi merepresentasikan arah dan besar dari komponen utama, menggunakan teknik komputasi *Singular Value Decomposition* (SVD)
4. Memilih komponen utama : mengurutkan nilai eigen dari yang terbesar lalu memilih k (dapat diatur) vektor eigen yang berkorespondensi dengan k nilai eigen terbesar
5. Proyeksikan data ke komponen utama : mengubah data dengan melakukan proyeksi terhadap subruang yang dibangun oleh komponen utama yang dipilih

Data-data yang diproyeksikan ini tidak menghilangkan informasi yang terdapat pada data aslinya sehingga cukup bagus digunakan untuk reduksi dimensi.

2.2.2. Metode Seleksi (fitur *scikit-learn*)

Scikit-learn menyediakan fitur seleksi pada Python Bernama *'SelectKBest'*. Serupa dengan PCA, fitur ini melakukan reduksi dimensi pada data yang diberikan. Namun, perbedaannya, fitur ini tidak menyadur informasi dari data-data yang tidak digunakan. Artinya, kolom data yang dianggap kurang berpengaruh betul-betul tidak digunakan. Maka dari itu, teknik ini sedikit menghilangkan informasi yang terdapat pada data aslinya. Terdapat beberapa fungsi skor bawaan, yaitu :

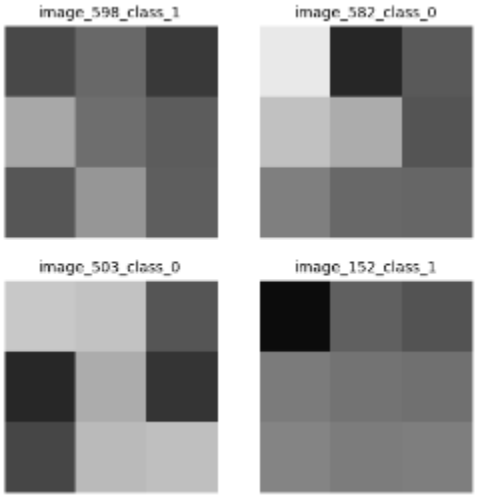
1. f_regression : digunakan untuk permasalahan regresi linear dan menghitung F-value
2. mutual_info_regression : digunakan untuk regresi dan menghitung informasi keterkaitan antara dua peubah acak
3. f_classif : digunakan untuk permasalahan klasifikasi dan menghitung ANOVA F-value

4. mutual_info_classif : digunakan untuk permasalahan klasifikasi dan menghitung informasi keterkaitan antara dua variabel
5. chi2 : digunakan untuk permasalahan klasifikasi dan menghitung chi-squared
6. SelectPercentile : digunakan untuk memilih beberapa persen fitur berdasarkan fungsi skor

Pada artikel ini, karena digunakan untuk tujuan klasifikasi dan mereduksi variabel-variabel yang tidak saling berkorelasi, digunakan fungsi skor mutual_info_classif.

Setelah dilakukan transformasi pada data aslinya menjadi variabel yang berjumlah kuadrat sempurna, data siap untuk dilakukan klasifikasi menggunakan CNN. Terdapat beberapa tahapan yang dilakukan untuk menggunakan model CNN, yaitu pembentukan gambar dari data transformasi, lalu masuk ke arsitektur CNN, dan terakhir adalah evaluasi.

Dari variabel-variabel yang sudah terpilih, setiap baris data akan diproses menjadi gambar. Pada hal ini, variabel dan urutan yang terpilih dapat mempengaruhi hasil dari prediksi mengingat nilai variabel dan urutannya mempengaruhi gambar yang dihasilkan. Gambar yang dihasilkan tersebut akan memiliki label masing-masing, sesuai dengan pelabelan yang ada. Setelah itu, gambar akan diproses pada arsitektur CNN yang akan dibangun. Berikut ini adalah contoh gambar yang dihasilkan



Setelah gambar sudah tersedia, dibangun arsitektur CNN untuk memproses gambar tersebut. Arsitektur CNN yang digunakan berasal dari paper *"Algorithmic financial trading with deep convolutional neural networks : Time series to image conversion approach"*. Terdapat 9 layer yang digunakan, yaitu : Input Layer ($m \times m$), dua buah *Convolutional Layers* ($m \times m \times 32$ dan $m \times m \times 64$), Max Pooling ($7 \times 7 \times 64$), dua buah dropout (0.25,0.50), *Fully Connected Layers* (128), dan *Output Layer* (2). Sebagai informasi, *Convolutional Layers* menggunakan operasi konvolusi, dengan cara perhitungan perhitungan sederhananya adalah

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a)$$

Karena dilakukan untuk gambar dua dimensi, operasi konvolusi diperluas menjadi

$$s(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-m, j-n)$$

Max Pooling yang dilakukan setelah *Convolutional Layers* bertujuan untuk membangun struktur *Deep Neural Network*. Setelah itu, *Fully Connected Layers* adalah tahapan dalam perhitungan bobot yang akan dicari. Terakhir, untuk mengeluarkan *output* yang dihasilkan, digunakan fungsi softmax yang dinilai sesuai dengan arsitektur CNN.

Arsitektur CNN ini diterapkan menggunakan Keras dan Tensorflow menggunakan epochs yang berbeda-beda agar dihasilkan prediksi yang akurat.

Selanjutnya, setelah dilakukan *training* dan *testing*, dilakukan evaluasi hasil arsitektur CNN yang dibentuk. Evaluasi dilakukan berdasarkan model secara komputasional. Hal ini dapat diukur menggunakan perhitungan akurasi dan skor F1 untuk masing-masing label target 0 dan 1.

3. Hasil

Berdasarkan skema yang telah dibangun, dilakukan beberapa simulasi dengan model yang berbeda-beda. Berikut ini adalah cuplikan dari data yang digunakan

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

Terdapat total 1025 data yang digunakan pada simulasi ini dengan kolom ‘age’ sampai ‘thal’ digunakan sebagai fitur dan kolom ‘target’ digunakan sebagai nilai yang akan diprediksi. Setelah itu, data tersebut digunakan pada model yang telah dibangun. Akan dibandingkan keseluruhan hasil dari model yang berbeda-beda. Setelah dilakukan model terbentuk, dilakukan uji untuk satu buah data berikut

```
age          50.0
sex           0.0
cp            1.0
trestbps    120.0
chol        244.0
fbs           0.0
restecg       1.0
thalach     162.0
exang         0.0
oldpeak       1.1
slope         2.0
ca            0.0
thal          2.0
target        1.0
```

3.1 Simulasi dengan MLP NumPy

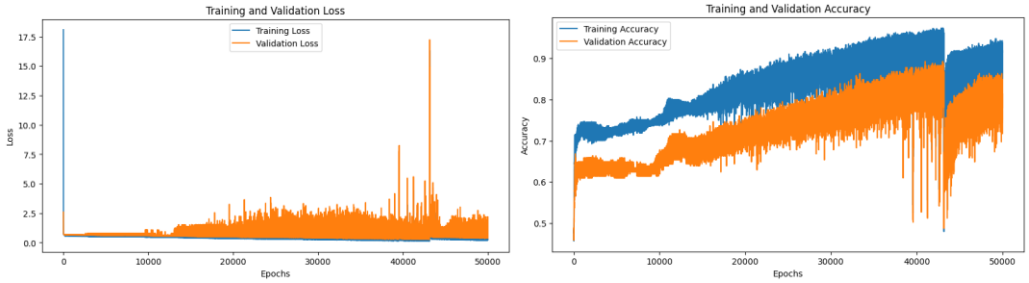
Pada simulasi ini, data diproses menggunakan MLP dengan *package* di NumPy. Dilakukan beberapa simulasi dengan mengubah *hyperparameter* untuk dibandingkan hasilnya.

- Simulasi Pertama :

  Dengan algoritma optimisasi adam dengan *learning rate* 0.01, loss function *binary cross entropy*, *random state* 999, dan konfigurasi arsitektur sebagai berikut

Layer	***Neuron***	***Activation Function***
Input Layer	13	*Relu*
Hidden Layer 1	128	*Relu*
Hidden Layer 2	64	*Sigmoid*
Output Layer	1	-

  diperoleh *train accuracy* 0.8975, *validation accuracy* 0.8292, dan *test accuracy* 0.7902 setelah 50000 epoch. Proses pembelajarannya adalah sebagai berikut.

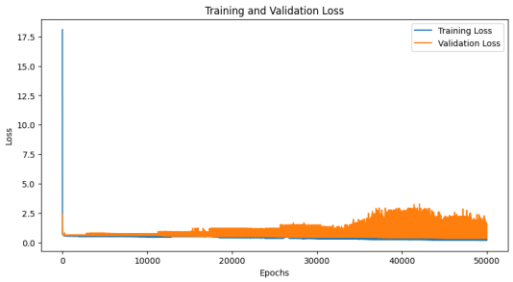


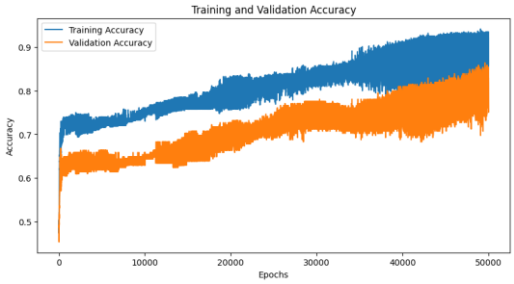
- Simulasi Kedua :

  Dengan algoritma optimisasi adam dengan *learning rate* 0.01, loss function *binary cross entropy*, *random state* 999, dan konfigurasi arsitektur sebagai berikut

Layer	***Neuron***	***Activation Function***
Input Layer	13	*Relu*
Hidden Layer 1	64	*Relu*
Hidden Layer 2	32	*Sigmoid*
Output Layer	1	-

diperoleh *train accuracy* 0.9170, *validation accuracy* 0.8243, dan *test accuracy* 0.7804 setelah 50000 epoch. Proses pembelajarannya adalah sebagai berikut.





3.2 Simulasi dengan MLP TensorFlow

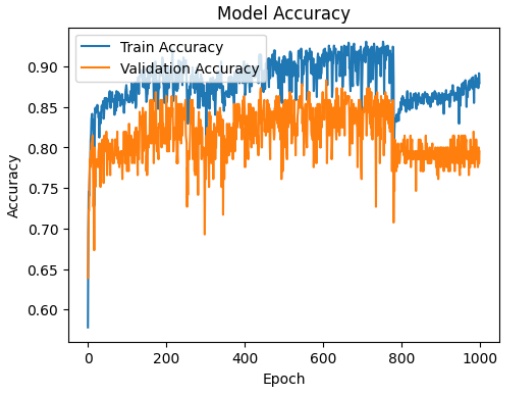
Pada simulasi ini, data diproses menggunakan MLP dengan *package* TensorFlow. Dilakukan beberapa simulasi dengan mengubah *hyperparameter* dan juga *layer* untuk dibandingkan hasilnya

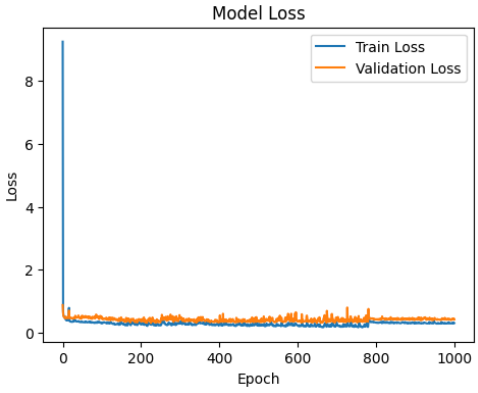
- Simulasi Pertama :

  Dengan algoritma optimisasi adam dengan *learning rate* 0.01, loss function *binary cross entropy*, *random state* 999, dan konfigurasi arsitektur sebagai berikut

Layer	***Neuron***	***Activation Function***
Input Layer	13	*Relu*
Hidden Layer 1	128	*Relu*
Hidden Layer 2	64	*Sigmoid*
Output Layer	1	-

diperoleh *accuracy* 0.8818, dan *val_accuracy* 0.7951 setelah 1000 *epoch*. Kemudian, proses pembelajarannya adalah sebagai berikut.

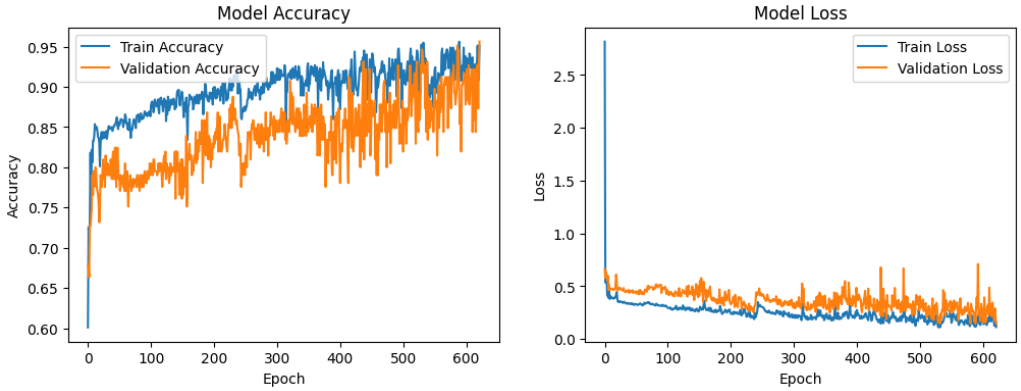




- Simulasi Kedua :
  Dengan algoritma optimisasi adam dengan *learning rate* 0.01, loss function *binary cross entropy*, *random state* 999, dan konfigurasi arsitektur sebagai berikut.

Layer	***Neuron***	***Activation Function***
Input Layer	13	*Relu*
Hidden Layer 1	64	*Relu*
Hidden Layer 2	32	*Sigmoid*
Output Layer	1	-

diperoleh *accuracy* 0.9445, dan *val_accuracy* 0.9561 setelah 622 *epoch*. Kemudian, proses pembelajarannya adalah sebagai berikut.



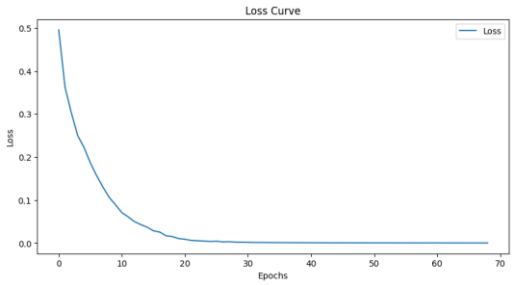
3.3 Simulasi dengan MLP Sklearn

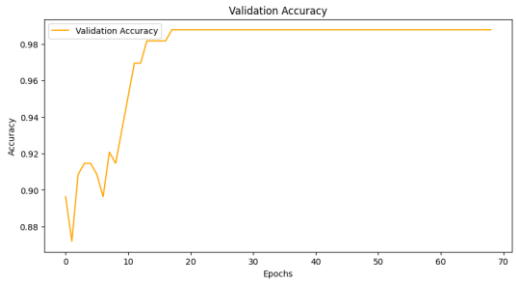
Pada simulasi ini, data diproses menggunakan MLP dengan *package* sklearn. Dilakukan beberapa simulasi dengan mengubah *hyperparameter* dan juga *layer* untuk dibandingkan hasilnya

- Simulasi Pertama :
  Dengan algoritma optimisasi adam dengan *learning rate* 0.01, loss function *cross entropy*, *random state* 999, dan konfigurasi arsitektur sebagai berikut

Layer	***Neuron***	***Activation Function***
Input Layer	13	*Relu*
Hidden Layer 1	128	*Relu*
Hidden Layer 2	64	*Sigmoid*
Output Layer	1	-

diperoleh *train accuracy* 0.9878, dan *validation accuracy* 0.9951 setelah 69 epoch. Proses pembelajarannya adalah sebagai berikut.

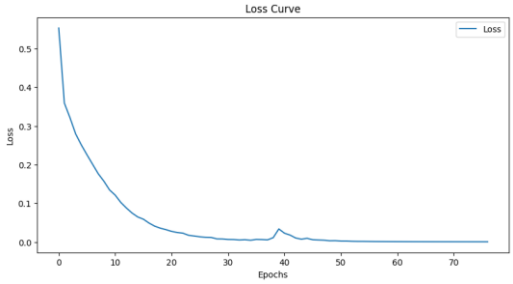


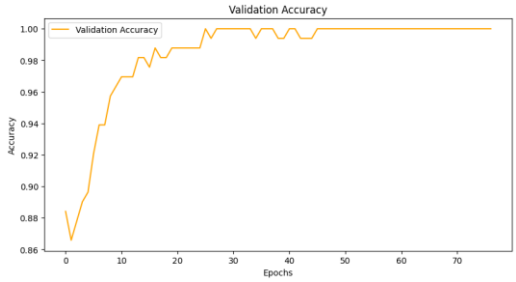


- Simulasi Kedua :
  Dengan algoritma optimisasi adam dengan *learning rate* 0.01, loss function *cross entropy*, *random state* 999, dan konfigurasi arsitektur sebagai berikut

Layer	***Neuron***	***Activation Function***
Input Layer	13	*Relu*
Hidden Layer 1	64	*Relu*
Hidden Layer 2	32	*Sigmoid*
Output Layer	1	-

diperoleh *train accuracy* 1.0, dan *validation accuracy* 1.0 setelah 77 epoch. Proses pembelajarannya adalah sebagai berikut.





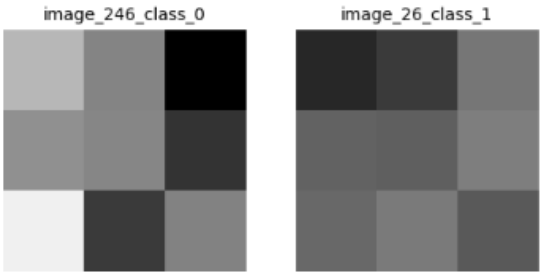
3.4 Simulasi dengan CNN Teknik Reduksi PCA

Pada simulasi ini, digunakan model CNN dengan teknik reduksi data PCA. Dimensi dari data akan diubah menjadi sebanyak kuadrat sempurna agar dapat dijadikan gambar berbentuk persegi. Berikut ini adalah hasil-hasil dari simulasi yang sudah dilakukan

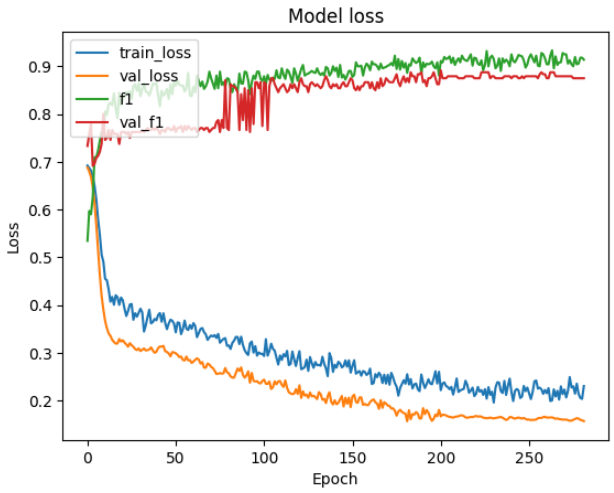
- Simulasi Pertama : Learning Rate 0.001
  Dari 13 fitur yang ada, dilakukan PCA untuk mereduksi data menjadi 9 dimensi. Setelah dilakukan hal tersebut, didapatkan

	0	1	2	3	4	5	6	7	8	target
0	-0.522556	-1.112803	0.956816	-1.149198	-0.559252	-1.505052	0.071292	0.049732	0.872570	0
1	2.590381	-0.533162	1.467315	1.536614	1.345335	1.524630	1.469460	0.594801	-0.127561	0
2	3.042352	-1.327521	-0.424765	1.567204	0.283814	-0.738182	0.378211	-1.397097	-0.836844	0
3	-0.492522	-0.276720	0.801442	-0.984277	-0.487587	-1.438634	0.385833	-1.566671	0.085219	0
4	2.187464	1.951477	-0.385539	0.295793	-2.386144	-0.563839	1.022689	1.682067	0.451377	0
...	...	...	...	...	...	...	...	...	...	...
1020	-0.762321	-0.512273	0.046672	-0.308011	-0.415454	0.083938	0.957084	-1.517729	-0.519319	1
1021	2.374273	-0.940859	0.182370	-0.628459	0.822406	0.373266	-0.493855	0.086828	-0.050143	0
1022	1.245073	-1.457356	-0.473873	-0.645240	-0.271197	1.786723	-0.623980	0.614532	-0.084198	0
1023	-1.620053	0.124443	-1.327956	-1.196804	-0.224913	1.263473	-0.498017	0.265437	0.345972	1
1024	0.934169	-1.778549	-0.005882	0.353372	-0.743381	-1.034172	-0.484996	-0.130881	-0.428642	0

yang dianggap merepresentasikan seluruh data asli yang dimiliki. Setelah itu, diambil persentasi untuk *training* dan *testing* sebesar 80% – 20%. Untuk setiap baris data, 9 fitur tersebut ditransformasikan menjadi gambar berikut



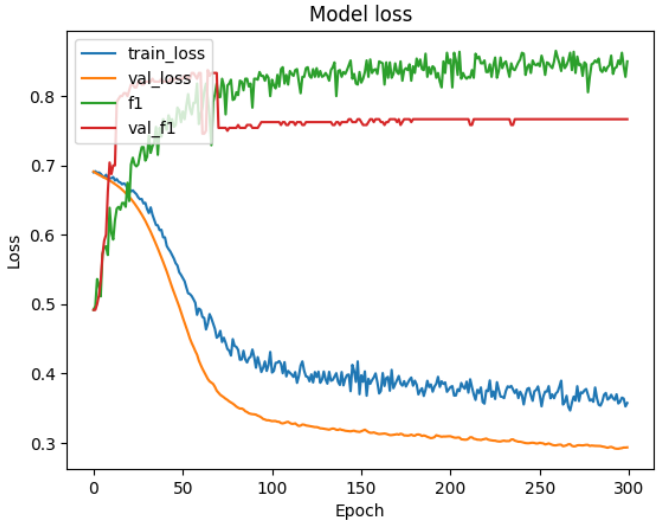
sehingga kelas gambar dengan target 0 adalah yang kiri dan kelas gambar dengan target 1 adalah yang kanan. Gambar tersebut kemudian diproses melalui model CNN yang telah dibangun dan digunakan epochs 300. Hasil yang didapatkan adalah sebagai berikut



Didapatkan bahwa pada setiap iterasi epoch *loss function* nya berkurang cukup cepat dan akurasinya meningkat. Secara umum, dilakukan evaluasi dan

didapatkan bahwa skor presisi kelas 0 adalah 0.96 dan presisi kelas 1 adalah 0.9. Sehingga, secara rata-rata presisi dari model ini adalah 0.93.

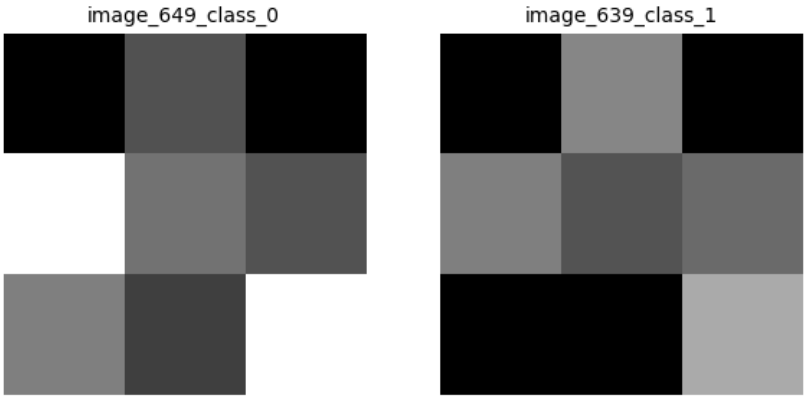
- Simulasi Kedua : Learning Rate 0.0001
  Untuk *learning rate* 0.0001 atau diperkecil dari sebelumnya, didapatkan hasil berikut



dengan evaluasi presisi kelas 0 adalah 0.74 dan kelas 1 adalah 0.8. Sehingga, presisi dari model ini adalah 0.77.

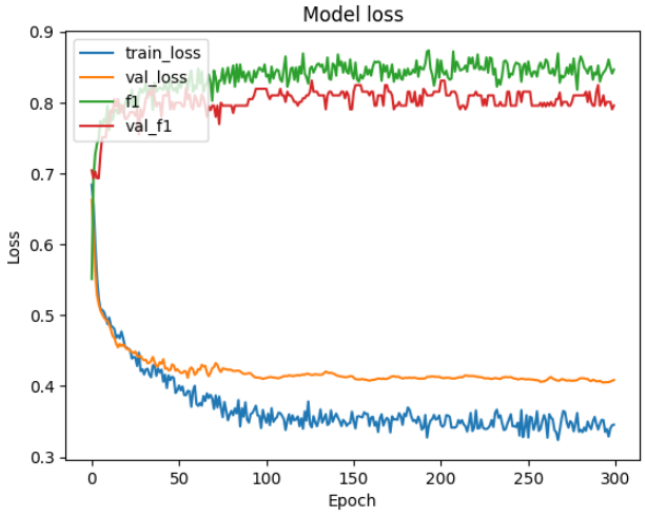
3.5 Simulasi dengan CNN Teknik Reduksi Seleksi

Pada simulasi ini, dilakukan reduksi dimensi data menjadi kuadrat sempurna menggunakan metode seleksi. Metode seleksi ini menyebabkan data asli yang digunakan utuh namun hanya dipilih variabel yang dianggap penting saja. Dari 13 fitur yang ada, dilakukan seleksi sebanyak 9 fitur sehingga dapat dijadikan gambar 3×3 yang akan diproses pada CNN. Gambar yang dihasilkan adalah sebagai berikut



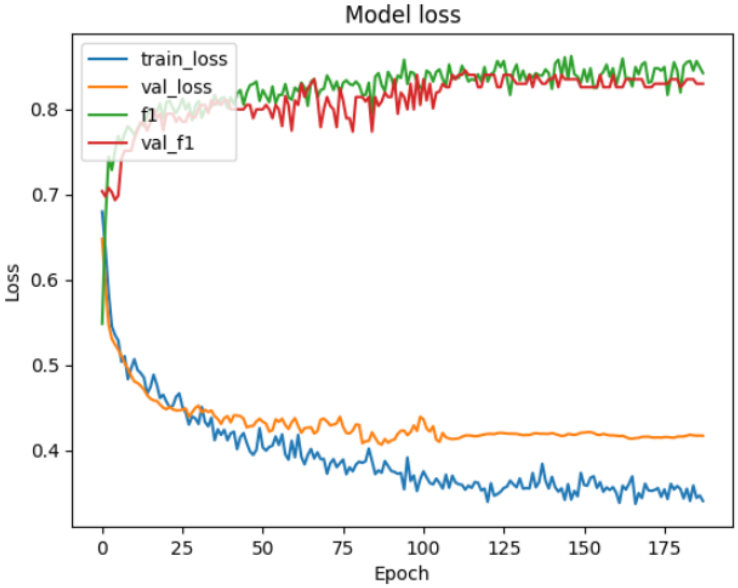
Setelah itu, gambar tersebut diproses menggunakan model CNN yang telah dibangun. Dilakukan dua kali simulasi dengan epochs yang berbeda

- Simulasi Pertama : Epochs 300
  Apabila digunakan epochs 300, didapatkan hasil berikut



Didapatkan bahwa presisi kelas 0 adalah 0.84 dan presisi kelas 1 adalah 0.78. Akibatnya, presisi model keseluruhan adalah 0.81.

- Simulasi Kedua : Epochs 500
  Apabila digunakan epochs 500, didapatkan hasil berikut



Dapat terlihat bahwa epoch berhenti di 188 karena model sudah tidak bisa meningkatkan performanya. Didapatkan bahwa presisi kelas 0 adalah 0.84 dan presisi kelas 1 adalah 0.78. Akibatnya, presisi model keseluruhan adalah 0.81

4. Diskusi

Dari keempat hasil simulasi di atas, didapat rangkuman berikut

Aspek	MLP NP		MLP TF		MLP SK		CNN PCA		CNN SL	
	1	2	1	2	1	2	1	2	1	2

Epochs	50.000	50.000	1000	622	69	77	300	300	300	500
LR	0.01	0.01	0.01	0.01	0.01	0.001	0.0001	0.001	0.001	0.01
Train Acc	0.8975	0.9170	0.8818	0.9445	0.9878	1	0.93	0.77	0.81	0.81

Dapat dilihat bahwa akurasi dari seluruh model yang dibangun sudah cukup baik, dengan nilai akurasi yang lebih dari 50%. Model dengan akurasi tertinggi adalah model MLP dengan TensorFlow atau Sklearn. Artinya, model-model tersebut sudah cukup dapat digunakan untuk melakukan klasifikasi dari data baru. Dengan perbandingan arsitektur model yang sama, kita dapat lihat bagaimana setiap *library* memberikan performanya dalam algoritma MLP. Misal, dengan tensorflow, kita dapatkan hasil dengan performa sekitar 80% setelah ratusan *epoch*. Selanjutnya untuk SK Learn, kita dapatkan akurasi lebih dari 90% dengan *epoch* 60-80 an. Kemudian, untuk numpy secara manual, kita peroleh hasil hingga 0.9 dengan epoch yang mencapai puluhan ribu. Hal ini berarti, kekonvergenan model untuk numpy lebih lemah daripada tensorflow dan SK Learn. Hal ini ditambah dengan pembelajaran pada numpy jauh lebih fluktuatif dibanding dengan pembelajaran pada SK Learn dan tensorflow.

Hasil prediksi yang dilakukan masing-masing model adalah sebagai berikut

Aspek	MLP NP		MLP TF		MLP SK		CNN PCA		CNN SL	
	1	2	1	2	1	2	1	2	1	2
Target	1	1	1	1	1	1	1	1	1	1

Terlihat bahwa seluruh model melakukan prediksi yang tepat sesuai dengan data. Dari hasil yang didapatkan tersebut, dilakukan analisis dan perbandingan terhadap masing-masing model.

Untuk model MLP, meskipun pada masa training model telah dilihat bahwa *library* untuk numpy lebih lemah dalam konvergensinya dibanding dengan *library* SK Learn dan tensorflow, tetapi secara umum semua model MLP yang telah dicoba mampu menangkap pola-pola yang terkandung dalam data. Hal ini dibuktikan dengan percobaan prediksi data yang menunjukan prediksi memiliki nilai 1.

Di lain sisi, terdapat dua buah model CNN yang dibangun. Dari hasil akurasi model, dapat terlihat bahwa akurasi tertinggi dapat dicapai oleh CNN dengan teknik reduksi PCA. Hal ini menunjukkan bahwa PCA dalam hal ini dapat merangkum seluruh data asli menjadi bentuk dimensi yang tereduksi tanpa menghilangkan informasi-informasi yang penting dari data. CNN menggunakan teknik seleksi memiliki akurasi yang cukup baik namun lebih rendah, hal ini dapat disebabkan karena informasi dari data yang diambil tidak dapat tertangkap secara keseluruhan. Terlalu banyak data atau informasi yang tidak digunakan sehingga model menjadi kesulitan dalam melakukan *fitting*.

Untuk kedua model CNN dan MLP, dapat disimpulkan bahwa model MLP lebih memiliki akurasi yang tinggi dibandingkan dengan menggunakan CNN. Hal ini dapat terjadi karena informasi data yang digunakan dari MLP adalah keseluruhan data. Akibatnya, data digunakan secara utuh (asli) sehingga tidak ada informasi yang hilang. Konstruksi CNN membutuhkan data yang memiliki fitur sejumlah kuadrat sempurna. Hal ini menyebabkan akurasi dari MLP lebih tinggi daripada yang dihasilkan CNN.

5. Kontribusi

Berikut ini adalah kontribusi yang dilakukan masing-masing anggota kelompok :

- Mochamad Fadlan Adhari (10120099) : Mencari data, membuat *syntax* CNN, membuat laporan
- Kevin Sean Hans Lopulala (10120074) : membuat *syntax* MLP numpy, Tensorflow, dan SK Learn