JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 3, Juli 2021, Page 1084-1089
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i3.3090

Optimasi *Hyperparameter TensorFlow* dengan Menggunakan Optuna di *Python*: Study Kasus Klasifikasi Dokumen Abstrak Skripsi

Siti Mujilahwati[1], Miftahus Sholihin[2], Retno Wardhani[3]

Program Studi Teknik Informatika, Fakultas Teknik, Universitas Islam Lamongan, Jawa Timur, Indonesia
Email: [1,*]moedjee@gmail.com, [2]miftahus.sholihin@gmail.com, [3]retzno@yahoo.com
Email Penulis Korespondensi: moedjee@gmail.com[1]

Abstrak−Pada era digital yang berkembang pesat saat ini peran komputasi pada kecerdasan buatan sangat dibutuhkan untuk dapat membantu para pelaku bisnis. Baik dari bidang perekonomian, Kesehatan dan juga Pendidikan. Penggunaan machine learning akan membantu pihak terkait dalam melihat, menganalisis dan mengambil keputusan. Dengan machine learning segala sesuatu permasalahan yang terkait dengan data akan dapat diselesaikan dengan cepat dan tepat. Permasalahannya adalah dokumen skripsi tiap tahun pasti semakin bertambah, akan menjadi sebuah dokumen yang sia-sia jika tidak dilakukan pengolahan data tersebut. Data skripsi yang sudah lampau dapat dijakian analisis dan pengambilan keputusan pada era skripsi berikutnya. Python merupakan salah satu Bahasa pemrograman yang populer digunakan untuk machine learning. Salah satu alasanya adalah banyaknya library yang berbasis python. Keras adalah salah satu library machine learning berbasis python. *TensorFlow* dapat digunakan jika berhubungan dengan pengolahan data yang banyak, termasuk data abstrak skripsi. Dengan demikian penelitian ini melakukan klasifikasi 140 dokumen abstrak skripsi dengan menggunakan keras-*TensorFlow* dengan tujuan bahwa berdasarkan isi abstrak tersebut akan diklasifikasikan pada 6 kelas yaitu Aplikasi Android, Data Mining, RPL, SPK, Pengolahan Citra Digital dan Sistem Pakar. Adapun hasil klasifikasi dengan data latih sebanyak 82 dokumen dengan setting model betch size = 12 dan epoch = 2 dengan nilai akurasi 89,04%. Sedangkan pada data uji test *loss* memiliki nilai lebih tinggi daripada nilai akurasi yang diperoleh 66,66%. Dengan memanfaatkan memaksimalkan kinerja *TensorFlow* dengan menambahkan parameter yang dimiliki scikit learn yaitu optuna. Data tes dilakukan optimalisasi dengan nilai trials sebanyak 500 akurasi meningkat menjadi 76.19%.

Kata Kunci: Optimasi; *TensorFlow*; Optuna; Python; Skripsi

Abstract−In today's rapidly growing digital era, the role of computing in artificial intelligence is needed to be able to help business people. Both in the fields of economy, health, and education. The use of machine learning will help related parties in viewing, analyzing, and making decisions. With machine learning, all problems related to data can be solved quickly and precisely. The problem is that the thesis document will increase every year, it will become a useless document if the data processing is not carried out. Past thesis data can be used for analysis and decision-making in the next thesis era. Python is one of the most popular programming languages used for machine learning. One reason is that there are many python-based libraries. Keras is a python-based machine learning library. *TensorFlow* can be used when dealing with large amounts of data processing, including thesis abstract data. Thus, this study classified 140 thesis abstract documents using hard-*TensorFlow* with the aim that based on the abstract content it would be classified into 6 classes, namely Android Applications, Data Mining, RPL, SPK, Digital Image Processing, and Expert Systems. The results of the classification with training data as many as 82 documents with model setting batch size = 12 and epoch = 2 with an *Accuracy* value of 89.04%. While the test *loss* test data has a higher value than the *Accuracy* value obtained by 66.66%. By utilizing maximizing *TensorFlow* performance by adding a parameter that Scikit Learn has, namely Optuna. The test data was optimized with a trial value of 500, the *Accuracy* increased to 76.19%.

Keywords: Optimization; *TensorFlow*; Optuna; Python; Thesis

1. PENDAHULUAN

Perkembangan kecerdasan buatan di era digital seperti ini dapat kita lihat dengan jelas bagaimana tumbuh dengan pesat dan sangat dibutuhkan. Baik dibidang Kesehatan, bisnis, dan juga Pendidikan. Skripsi mahasiswa, semakin tahun semakin bertingkat. Akan menjadi suatu dokumen yang sia-sia jika tidak dilakukan klasifikasi. Terutama dengan perkembangan teknologi machine learning seperti saat ini. Dengan melakukan klasifikasi dokumen skripsi, baik berdasarkan teks abstrak maka dapat diperoleh hasil analisis dari dokumen tersebut berdasarkan tujuannya. Misal analisis masalah topik skripsi mahasiswa, yang artinya akan dilakukan klasifikasi dokumen skripsi berdasarkan topik. Dapat menganalisis trend topik skripsi, dapat melakukan kecocokan abtrak dengan topik. Dan masih banyak lagi yang dapat dilakukan dengan dokumen skripsi tersebut.

Deep learning merupakan bagian dari machine learning yang dapat mengatasi pada kasus data yang besar dan semakin meningkat. Beberapa klasifikasi dokumen yang telah dilakukan pada penelitian sebelumnya menyatakan bahwa *Deep learning* dapat bekerja dengan baik[1], [2]. Selain itu *Deep learning* dapat bekerja dengan baik karena memiliki konsep bekerja secara mendalam seperti otak manusia. Klasifikasi merupakan pengelompokkan berdasarkan kelas target yang telah dilabeli sebelumnya. Beberapa penelitian yang melakukan klasifikasi dengan *Deep learning* lebih banyak menggunakan data gambar[3], [4]. Sedangkan klasifikasi teks dengan memanfaatkan *Deep learning* sering digunakan menganalisis sentiment dengan menggunakan data short teks baik dari Bahasa Indonesia, arab dan Bahasa inggris[5]–[9]. Ada beberapa algoritma *Deep learning* yang dapat diterapakan untuk melakukan klasifikasi gambar maupun teks diantaranya Artificial *Neural Network*[10], Convolutional *Neural Network*[11], [12], Backpropagation *Neural Network*[13]. Dari beberapa algoritma tersebut dapat memberikan hasil yang baik, akan tetapi ada beberapa penelitian sebelumnya juga melakukan

Submitted: **25/06/2021**; Accepted: **22/07/2021**; Published: **31/07/2021**

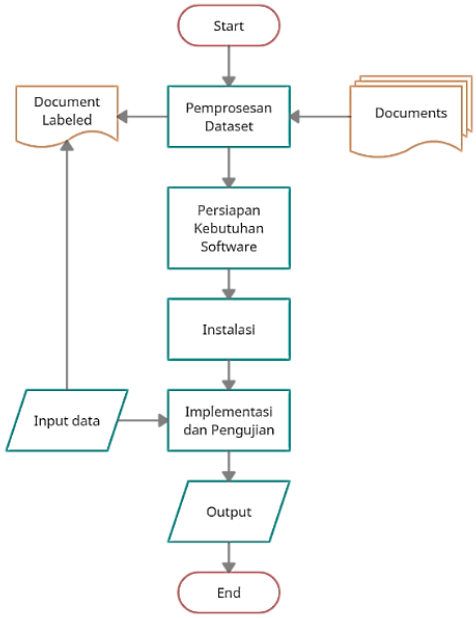
JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 3, Juli 2021, Page 1084-1089
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i3.3090

optimasi dari algoritma yang digunakan seperti yang dilakukan pada optimasi algoritma *Neural Network* dengan algoritma genetika [14]–[18].

Dari uraian di atas serta mempelajari beberapa penelitian yang telah disebutkan, maka penelitian ini menemukan start of the art. Yaitu penelitian ini akan melakukanklasifikasi dokumen abstrak sekripsi dengan menggunakan konsep *Deep learning* dengan memanfaatkan kerangka *TensorFlow* dari data teks. Dimana pada penelitian sebelumnya banyak yang menggunakan data gambar dan sort teks. Selain itu optimasi hasil akurasi klasifikasi akan mencoba memanfaatkan *library optuna di scikit learning* pada *python*.

2. METODOLOGI PENELITIAN

Desain alur dari penelitian ini adalah digambarkan pada gambar 1 berikut ini.



Gambar 1. Alur Penelitian

Konsep penelitian yang bersifat pengujian dan analisis pada penelitian ini, maka pertama yang dilakukan adalah:

1) Analisis Data

Data yang akan digunakan adalah data atau dokumen abstrak skripsi mahasiswa Teknik informatika lulusan 2020, dokumen yang tersimpan sebanyak 103 file. Setiap dokumen akan dilabeli dengan kelas berdasarkan topik yang dibahas. Setelah dibentuk ada 6 kelas target, dan masing-masing memiliki sebaran seperti ditampilkan pada tabel 1 berikut ini.

Tabel 1. Data Faktual Masing-masing Kelas

No	Kelas	Jumlah Dokumen
1	Aplikasi Android	38
2	Data Mining	19
3	RPL	16
4	SPK	14
5	Pengolahan Citra Digital	8
6	Sistem Pakar	8
Total		103

Data tersebut nantinya akan dilakukan pengujian klasifikasi pada masing-masing kelas dengan model *Deep learning* menggunakan library Keras *TensorFlow* di *python*.

2) Persiapan kebutuhan software dan library

a) Jupyter Notebook

Dalam hal ini, "notebook" atau "dokumen notebook" menunjukkan dokumen yang berisi elemen kode dan seperti teks, gambar, tautan, persamaan. Karena campuran elemen kode dan teks, dokumen ini adalah tempat yang ideal untuk menyatukan deskripsi analisis, dan hasilnya, serta dapat dieksekusi melakukan analisis data secara real time.[19]. Jupiter Notebook merupakan aplikasi edit teks untuk machine learning dan bersifat client-server. Aplikasi ini dapat menjalankan kode *python*.

b) *Python*

Submitted: **25/06/2021**; Accepted: **22/07/2021**; Published: **31/07/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 3, Juli 2021, Page 1084-1089
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i3.3090

Python merupakan Bahasa pemrograman yang berisifat interaktif dan dapat dijalankan diberbagai macam platform atau aplikasi. Salah satunya adalah aplikasi Jupyter Notebook. Dengan menggunakan aplikasi tersebut maka kita dapat langsung mengeksekusi setiap baris kode *python*. Banyak dukungan library apabila kita membuat program *python* untuk pengolahan data maupun machine learning. Pada penelitian ini menggunakan *python* versi 3.0.

c) Keras – *TensorFlow*

Keras merupakan API pada *Deep learning* pada pemprograman *python* dan berjalan di atas machine learning *TensorFlow*. Keras memiliki ciri :

Simple : tapi tidak sederhana. Keras mengurangi beban kognitif pengembang untuk membebaskan Anda untuk fokus pada bagian masalah yang benar-benar penting.

Fleksibel : Keras mengadopsi prinsip pengungkapan progresif kompleksitas: alur kerja sederhana harus cepat dan mudah, sementara alur kerja lanjutan yang sewenang-wenang harus dimungkinkan melalui jalur yang jelas yang dibangun berdasarkan apa yang telah Anda pelajari.

Powerful : Keras memberikan kinerja dan skalabilitas kekuatan industri: digunakan oleh organisasi dan perusahaan termasuk NASA, YouTube, atau Waymo.

Cara memanggil Keras di *python* model *Neural Network* :

from TensorFlow import keras
layers = keras.layers
models = keras.models

pada penelitian ini menggunakan *TensorFlow* versi 2.5.0

d) SCKIT LEARN – Optuna

Installasi optuna : $pip install optuna

Parameter *tuning* model *Neural Network* yang digunakan pada optuna disajikan pada Tabel 2 berikut ini.

Tabel 2. Parameter Fungsi Objektif Optuna

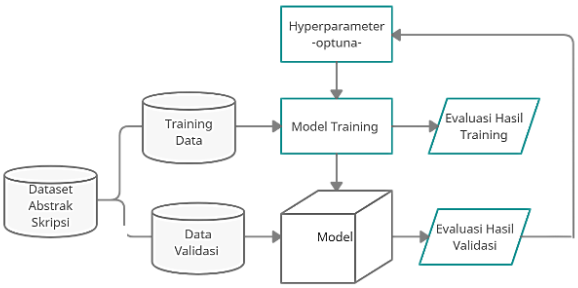
No	Deskripsi Parameter	Kode	Rentang Nilai
1	Banyak Layer	Layer	[1..2]
2	Banyak Neuron di hidden layer 1	h1	[10…500]
3	Banyak Neuron di hidden layer 2	h2	[10…500]
4	Iterasi Maksimal	i	[10…1000]
5	Inisial Learning Rate	ir	[0.00001…0.1]
6	`n_iter_no_change`	ni	[10…30]

Dari parameter dan rentang nilai yang akan digunakan pada penelitian ini, dan selanjutnya akan di eksekusi sebagai nilai fungsi objektif pada Optuna.

3) Desain Pengujian dan Analisis

Kerangka alur penelitian ini adalah digambarkan pada Gambar 1 berikut ini. Model optimasi *hyperparameter*.

Adapun alur dari *hyperparameter* yang dibangun dapat dilihat seperti pada Gambar 2 berikut ini.



Gambar 2. Alur Klasifikasi dan *Hyperparameter Tuning*

Data dokumen abstrak yang telah dilabeli sesuai dengan tema kelas, selanjutnya akan dilakukan *splitting* antara data latih dan data uji. Konsep *Neural Network* apabila nilai *loss* lebih renda dari nilai akurasi maka hasil dianggap baik. Apabila nilai *loss* lebih tinggi disbanding akurasi maka diperluka optimasi dengan optuna. Pemodela *Neural Network* dibuat dengan library Keras-*TensorFlow*.

3. HASIL DAN PEMBAHASAN

Hasil penelitian untuk klasifikasi model *Deep learning* dengan algoritma *Neural Network* dengan memanfaatkan pemodelan *TensorFlow* mendapatkan hasil akurasi klasifikasi cukup baik. Pengujian dan validasi digunakan

Submitted: **25/06/2021**; Accepted: **22/07/2021**; Published: **31/07/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 3, Juli 2021, Page 1084-1089
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i3.3090

manual *tuning*. Akan tetapi dengan penambahan otomatisasi *tuning* dengan optuna yang dilakukan dengan model *hyperparameter*. Nilai akurasi menjadi meningkat. Berikut pembahasannya.

```
In [9]: abstrak['Kelas'].value_counts()

Out[9]: Aplikasi_Android           38
        Data_Mining                19
        RPL                        16
        SPK                        14
        Pengolahan_Citra_Digital    8
        Sistem_Pakar                8
        Name: Kelas, dtype: int64
```

Gambar 3. *Dataset*

Selanjutnya adalah membangun data latih dengan model split. Data latih yang digunakan secara acak dengan rasio 0.8 dari *dataset*. Adapun Teknik split yang digunakan ditampilkan pada Gambar 4 berikut ini.

```
In [11]: train_size = int(len(abstrak) * .8)
         print ("Train size: %d" % train_size)
         print ("Test size: %d" % (len(abstrak) - train_size))

         Train size: 82
         Test size: 21
```

Gambar 4. *Dataset*

Data yang diperoleh dari pembagian secara split adalah seperti Nampak pada Gambar 4 di atas. Data latih atau train sebesar 80% = 82 dokumen. Sedangkan data uji atau Test = 21 dokumen. Selanjutnya membangun model training proses build Keras *TensorFlow*. Sebagau berikut.

```
In [24]: #Mulai membangun model training
         # Memulai proses build model
         model = models.Sequential()
         model.add(layers.Dense(512, input_shape=(max_words,)))
         model.add(layers.Activation('relu'))
         model.add(layers.Dense(num_classes))
         model.add(layers.Activation('softmax'))

         model.compile(loss='categorical_crossentropy',
                       optimizer='adam',
                       metrics=['accuracy'])
```

Gambar 5. Pemodelan TensoFlow *Neural Network*

Melakukan validasi data latih menggunakan batch_size = 12, *epochs* = 2, drop_ratio = 0.5 dengan memperoleh hasil *loss* sebesar 0.6721 dan *Accuracy* sebesar 0.9589. dari validasi data latih tersebut dan selanjutnya dilakukan validasi data uji atau *testing* diperoleh hasil *loss* sebesar 1.0442 dan *Accuracy* sebesar 0.6190. hasil dapat dilihat pada Gambar 6 berikut ini.

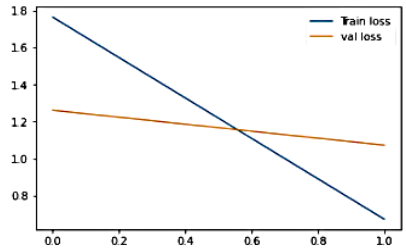
```
In [27]: history = model.fit(x_train, y_train, batch_size=batch_size,epochs=epochs,verbose=1,validation_split=0.1)

         Epoch 1/2
         7/7 [==============================] - 1s 31ms/step - loss: 1.7647 - accuracy: 0.3014 - val_loss: 1.2614 - val_accuracy: 0.4444
         Epoch 2/2
         7/7 [==============================] - 0s 7ms/step - loss: 0.6721 - accuracy: 0.9589 - val_loss: 1.0725 - val_accuracy: 0.4444

In [28]: # Mulai melakukan tindakan evaluasi terhadap hasil akurasi yang dimiliki oleh model yang sudah dilakukan training
         score = model.evaluate(x_test, y_test,
                                batch_size=batch_size, verbose=1)
         print('Test loss:', score[0])
         print('Test accuracy:', score[1])

         2/2 [==============================] - 0s 8ms/step - loss: 1.0443 - accuracy: 0.6190
         Test loss: 1.0442545413970947
         Test accuracy: 0.6190476417541504
```

Gambar 6. Hasil Validasi *Train* dan *Test*




```
<Figure size 432x288 with 0 Axes>
```

Gambar 7. Visualisasi Train dan Test Validasi

Submitted: **25/06/2021**; Accepted: **22/07/2021**; Published: **31/07/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 3, Juli 2021, Page 1084-1089
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i3.3090

Karena hasil *Accuracy* yang diperoleh tidak maksimal selanjutnya dilakukan pemodelan optimasi dengan scikit Learn model optuna. Implementasi di *python* adalah seperti terlihat pada gambar 8 berikut ini.

```
In [34]: def objective(trial):
             layers = trial.suggest_categorical('layers',[1,2])
             h1 = trial.suggest_int('h1',10,500)
             i = trial.suggest_int('max_i',10,1000)
             lr = trial.suggest_uniform('lr',0.00001,0.1)
             ni = trial.suggest_int('noi',10,30)

             if layers==1:
                 clf = MLPClassifier(random_state=11,
                         hidden_layer_sizes=(h1),
                         max_iter=i, learning_rate_init=lr,
                         n_iter_no_change=ni)
             else:
                 h2 = trial.suggest_int('h2',10,500)
                 clf = MLPClassifier(random_state=11,
                         hidden_layer_sizes=(h1,h2),
                 max_iter=i, learning_rate_init=lr, n_iter_no_change=ni)

             clf.fit(x_train, y_train)

             return clf.score(x_test, y_test)

         study = optuna.create_study(direction='maximize')
         study.optimize(objective, n_trials=500)
         print(study.best_params)
[I 2021-06-20 07:26:30,739] Trial 491 finished with value: 0.47619047619047616 and parameters: {'layers': 2, 'h1': 398, 'max_
i': 364, 'lr': 0.021057086988342202, 'noi': 12, 'h2': 404}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:32,674] Trial 492 finished with value: 0.6666666666666666 and parameters: {'layers': 2, 'h1': 400, 'max_
i': 420, 'lr': 0.027155985040548045, 'noi': 12, 'h2': 414}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:35,386] Trial 493 finished with value: 0.38095238095238093 and parameters: {'layers': 2, 'h1': 489, 'max_
i': 630, 'lr': 0.03331849696425122, 'noi': 15, 'h2': 435}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:39,212] Trial 494 finished with value: 0.6190476190476191 and parameters: {'layers': 2, 'h1': 476, 'max_
i': 608, 'lr': 0.032263440431374794, 'noi': 13, 'h2': 383}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:39,682] Trial 495 finished with value: 0.38095238095238093 and parameters: {'layers': 1, 'h1': 372, 'max_
i': 497, 'lr': 0.037265065383581916, 'noi': 12}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:41,891] Trial 496 finished with value: 0.38095238095238093 and parameters: {'layers': 2, 'h1': 478, 'max_
i': 624, 'lr': 0.03169142305691128, 'noi': 12, 'h2': 390}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:43,552] Trial 497 finished with value: 0.5238095238095238 and parameters: {'layers': 2, 'h1': 493, 'max_
i': 599, 'lr': 0.03289786618696862, 'noi': 11, 'h2': 380}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:44,275] Trial 498 finished with value: 0.47619047619047616 and parameters: {'layers': 2, 'h1': 371, 'max_
i': 441, 'lr': 0.009230489446432028, 'noi': 12, 'h2': 400}. Best is trial 340 with value: 0.7619047619047619.
[I 2021-06-20 07:26:45,493] Trial 499 finished with value: 0.47619047619047616 and parameters: {'layers': 2, 'h1': 499, 'max_
i': 467, 'lr': 0.016147233165809047, 'noi': 15, 'h2': 416}. Best is trial 340 with value: 0.7619047619047619.

{'layers': 2, 'h1': 396, 'max_i': 397, 'lr': 0.015662186386458686, 'noi': 13, 'h2': 428}
```

Gambar 8. Model dan Hasil Optuna

Setelah menambahkan *hyperparameter* untuk optimasi secara otomatis dapat dilihat hasil *Accuracy* meningkat menjadi 0.7619 = 76.19% dengan 'layers': 2, 'h1': 396, 'max_i': 397, 'lr': 0.015662186386458686, 'noi': 13, 'h2': 428. Menggunakan n_trial sebanyak 500 kali. Dengan demikian dapat simpulkan bahwa penambahan *hyperparameter* dengan model optuna dapat menambah atau meningkatkan akurasi secara otomatis tanpa harus melakukan setting epoch dan betc Size pada pemodelan di *TensorFlow*.

4. KESIMPULAN

Dari hasil pengujian klasifikasi atau validasi klasifikasi dokumen abstrak skripsi dengan menggunakan Keras-*TensorFlow* di *python* maka disimpulkan bahwa model *Neural Network* yang diterapkan pada *TensorFlow* dengan menggunakan betch size sebesar 12 dan 2 epoch diperoleh nilai validasi sebesar 66.66%. Selanjutnya dengan menambahkan *hyperparameter* otomatisasi meningkat dengan nilai hasil *Accuracy* sebesar 76.19%.

REFERENCES

[1] A. D. Wattimena, "Analisis Sentimen Teks Bahasa Indonesia Pada Media Sosial Menggunakan Algoritma Convolutional *Neural Network* (Studi Kasus : E-Commerce)," p. 248, 2018.

[2] D. Yuliana and C. Supriyanto, "Klasifikasi Teks Pengaduan Masyarakat Dengan Menggunakan Algoritma *Neural Network*," p. 27, 2019.

[3] A. Peryanto, A. Yudhana, and R. Umar, "Rancang Bangun Klasifikasi Citra Dengan Teknologi *Deep learning* Berbasis Metode Convolutional *Neural Network*," *Format J. Ilm. Tek. Inform.*, vol. 8, no. 2, p. 138, Feb. 2020, doi: 10.22441/format.2019.v8.i2.007.

[4] Slamet Fifin Alamsyah, "Implementasi *Deep Learning* Untuk Klasifikasi Tanaman Toga Berdasarkan Ciri Daun Berbasis Android," *Ubiquitous Comput. Its Appl. J.*, pp. 113–122, Dec. 2019, doi: 10.51804/ucaiaj.v2i2.113-122.

[5] P. Cen, K. Zhang, and D. Zheng, "Sentiment Analysis Using *Deep learning* Approach," *J. Artif. Intell.*, vol. 2, no. 1, pp. 17–27, 2020, doi: 10.32604/jai.2020.010132.

[6] K. Devipriya, D. Prabha, V. Pirya, and S. Sudhakar, "*Deep learning* Sentiment Analysis For Recommendations In Social Applications," vol. 9, no. 01, p. 4, 2020.

[7] O. Habimana, Y. Li, R. Li, X. Gu, and G. Yu, "Sentiment analysis using *Deep learning* approaches: an overview," *Sci. China Inf. Sci.*, vol. 63, no. 1, p. 111102, Jan. 2020, doi: 10.1007/s11432-018-9941-6.

[8] A. Hassan and A. Mahmood, "*Deep learning* approach for sentiment analysis of short texts," in *2017 3rd International Conference on Control, Automation and Robotics (ICCAR)*, Nagoya, Japan, Apr. 2017, pp. 705–710. doi: 10.1109/ICCAR.2017.7942788.

[9] M. Heikal, M. Torki, and N. El-Makky, "Sentiment Analysis of Arabic Tweets using Deep Learning," *Procedia Comput. Sci.*, vol. 142, pp. 114–122, 2018, doi: 10.1016/j.procs.2018.10.466.

Submitted: **25/06/2021**; Accepted: **22/07/2021**; Published: **31/07/2021**

JURNAL MEDIA INFORMATIKA BUDIDARMA
Volume 5, Nomor 3, Juli 2021, Page 1084-1089
ISSN 2614-5278 (media cetak), ISSN 2548-8368 (media online)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/mib
DOI 10.30865/mib.v5i3.3090

[10] R. R. Santoso, R. Megasari, and Y. A. Hambali, "Implementasi Metode Machine Learning Menggunakan Algoritma Evolving Artificial *Neural Network* Pada Kasus Prediksi Diagnosis Diabetes," p. 13.
[11] H. Juwiantho, E. I. Setiawan, J. Santoso, and M. H. Purnomo, "Sentiment Analysis Twitter Bahasa Indonesia Berbasis Word2vec Menggunakan Deep Convolutional *Neural Network*," p. 8.
[12] B. P. Putra, B. Irawan, and S. Si, "Deteksi Ujaran Kebencian Dengan Menggunakan Algoritma Convolutional *Neural Network* Pada Gambar," p. 8.
[13] A. M. Zamani and J. A. R. Hakim, "Implementasi Algoritma Genetika pada Struktur Backpropagation *Neural Network* untuk Klasifikasi Kanker Payudara," vol. 1, no. 1, p. 6, 2012.
[14] I. Ali and L. Sularto, "Optimasi Parameter Artificial *Neural Network* Menggunakan Algoritma Genetika Untuk Prediksi Kelulusan Mahasiswa," *J. ICT Inf. Commun. Technol.*, vol. 18, no. 1, pp. 54–59, Aug. 2019, doi: 10.36054/jict-ikmi.v18i1.52.
[15] M. Badrul, "Optimasi *Neural Network* Dengan Algoritma Genetika Untuk Prediksi Hasil Pemilukada," . *P*, p. 14.
[16] R. Dina, "Optimasi Backpropagation *Neural Network* Menggunakan Metode Algoritma Genetika Dalam Memprediksi Jumlah Pengangguran," p. 149.
[17] Electrical Engineering Universitas Udayana, N. Widiangga Gautama, A. Dharma, and M. Sudarma, "Analisis Metode Rbf-Nn Dengan Optimasi Algoritma Genetika Pada Peramalan Mata Uang Eur/Usd," *Maj. Ilm. Teknol. Elektro*, vol. 15, no. 2, pp. 107–114, Dec. 2016, doi: 10.24843/MITE.1502.16.
[18] A. Kusnadi and J. Pratama, "Implementasi Algoritma Genetika dan *Neural Network* Pada Aplikasi Peramalan Produksi Mie (Studi Kasus : Omega Mie Jaya)," p. 5.
[19] K. Willems, "Jupyter Notebook Tutorial: The Definitive Guide." https://www.datacamp.com/community/tutorials/tutorial-jupyter-notebook

Submitted: **25/06/2021**; Accepted: **22/07/2021**; Published: **31/07/2021**