Jurnal Politeknik Caltex Riau
Terbit Online pada laman https://jurnal.pcr.ac.id/index.php/jkt/
| e- ISSN : 2460-5255 (Online) | p- ISSN : 2443-4159 (Print) |

Implementasi Metode LightGBM Untuk Klasifikasi Kondisi Abnormal Pada Pengemudi Sepeda Motor Berbasis Sensor *Smartphone*

R Rizki Rachmadi[1], Amang Sudarsono[2] dan Tri Budi Santoso[3]

[1]Politeknik Elektronika Negeri Surabaya, Teknik Elektro, email: rizkirachmadi@pasca.student.pens.ac.id
[2]Politeknik Elektronika Negeri Surabaya, Teknik Telekomunikasi, email: amang@pens.ac.id
[3]Politeknik Elektronika Negeri Surabaya, Teknik Telekomunikasi, email: tribudi@pens.ac.id

Abstrak

Kecelakaan lalu lintas merupakan salah satu penyebab angka kematian yang cukup tinggi Dengan kondisi demografis di Indonesia, di mana pengendara sepeda motor adalah tipe yang mendominasi lalu lintas jalan raya, sehingga resiko tertimpa kecelakaan lalu lintas leboh tinggi dibanding pengendara lain. Sistem deteksi aktivitas pada kendaraan bermotor yang telah banyak dibangun umumnya terfokus pada pengemudi mobil, dan memiliki masalah utama di waktu komputasi yang tinggi. Untuk mengatasi permasalahan ini, dalam penelitian kali ini, dibuat suatu sistem deteksi aktivitas abnormal dari pengendara sepeda motor dengan menggunakan metode Light Gradient Boosting Machine (LightGBM). Sistem tersebut didesain untuk memiliki waktu komputasi yang rendah dan dapat menghasilkan respons yang cepat terhadap perubahan gerakan yang terjadi dalam kecepatan tinggi. Untuk melakukan proses pelatihan model LightGBM, akan digunakan data yang berasal dari sensor Accelerometer dan Gyroscope yang tedapat pada smartphone, yang akan digunakan untuk mendeteksi gerakan yang dilakukan oleh seorang pengendara. Model yang didapat dari proses pelatihan dengan menggunakan data yang telah dikumpulkan menunjukkan tingkat akurasi setinggi 82% pada pengetesan menggunakan data yang telah disiapkan, dan menunjukkan akurasi hampir 70% dalam proses deteksi secara real-time, dengan waktu komputasi 10 mili detik, membuktikan bahwa sistem yang didesain bekerja 5 kali lipat lebih cepat dibanding sistem yang telah ada .

Kata kunci: *Kecelakaan Lalu Lintas, LightGBM, Waktu Komputasi*

Abstract

Traffic accident is one of the most significant contributors which makes the death number is increasing around the world. With the demographic condition from Indonesia, motorcycle driver is the types of the driver that dominated the traffic, therefore increasing the probability of caught in a traffic accident The existing Vehicle Activity Detection System (VADS) mainly focused on the car driver, with the main problem is that the computational time from the system is too high to be implemented on a real-time condition. To solve this problem, in this research, a classification system for abnormal driving behavior from motorcycle drivers is created, using Light Gradient Boosting Machine (LightGBM) model. The system is designed to be lightweight in computation and very fast in response to the changes of the activities with a high velocity. To train the LightGBM model, the data from Accelerometer and Gyroscope sensor, that has been integrated

Dokumen diterima pada 24 Oktober, 2021
Dipublikasikan pada 24 Desember, 2021

into a smartphone, will be used to detect the movement from a driver. The accuracy rate from the proposed model is reaching 82% on the test dataset and shows a promising result of around 70% on the real-time detection process. With a computational time of around 10ms, the proposed system is able to work 5 times faster than the existing system.

Keywords: *Traffic Accident, LightGBM, Computational Time*

1. Pendahuluan

Dalam satu dekade terakhir, kecelakaan lalu lintas dan keamanan jalan adalah salah satu masalah kritis yang sedang dihadapi oleh berbagai negara di dunia. Hal ini juga terjadi di Indonesia, dimana angka kecelakaan lalu lintas mencapai angka 28.000, dan rasio kematian yang disebabkan olehnya adalah 12 per 100.000 penduduk [1]. Angka ini sangat tinggi dibandingkan dengan negara-negara tetangga Indonesia, seperti Singapore dengan rasio 4.8 dan Australia dengan 5.8. Salah satu faktor yang mempengaruhi tingginya angka tersebut adalah perkembangan yang pesat di bidang teknololgi, membuat lonjakan yang tinggi dalam proses produksi kendaraan bermotor. Dari semua jenis kendaraan bermotor, pengendara sepeda motor adalah tipe yang paling sering terkena kecelakaan lalu lintas, dimana angka kecelakaan pada pengendara sepeda motor lebih tinggi 200% dibanding dengan pengendara mobil. Hal ini memiliki hubungan yang linear dengan banyaknya pengguna sepeda motor di Indonesia, yang hampir mencapai 150 juta pengendara pada tahun 2015 dan terus berkembang tiap tahunnya.

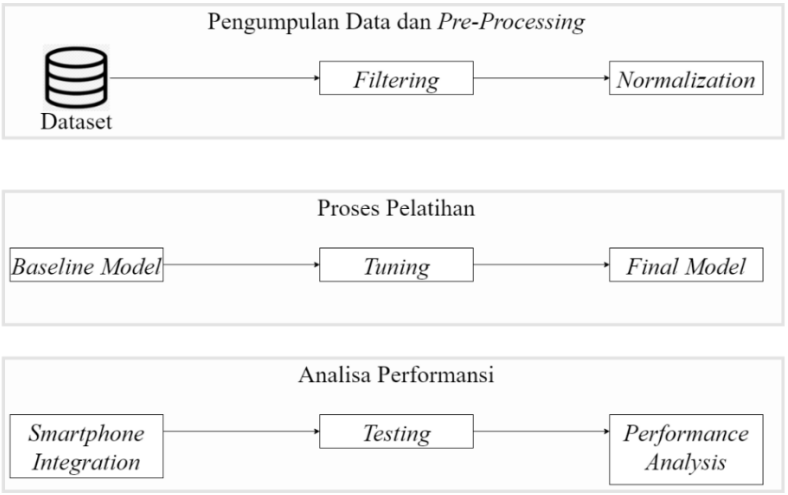
Dikarenakan tingginya angka kematian yang disebabkan oleh kecelakaan lalu lintas tersebut, pembuatan sistem deteksi aktivitas pengemudi merupakan salah satu topik penelitian yang sering digunakan pada penelitian-penelitian akhir-akhir ini. Salah satu metode yang sering digunakan adalah dengan menggunakan model *Deep Learning*, dimana fitur yang paling umum digunakan adalah fitur wajah dari pengemudi tersebut[2], atau dengan menggunakan bantuan sensor-sensor yang terdapat pada suatu *smartphone*[3]. Sistem yang menggunakan metode tersebut menunjukkan performa yang cukup menjanjikan, namun terdapat 2 kelemahan dari sistem-sistem tersebut. Yang pertama, metode *Deep Learning* membutuhkan konsumsi daya komputasi yang tinggi, dimana waktu komputasi akan meningkat secara signifikan apabila peralatan yang digunakan pada saat implementasi tidak memiliki daya komputasi yang cukup. Yang kedua, sistem-sistem tersebut secara umum dikembangkan untuk mendeteksi aktivitas pada pengemudi mobil. Dengan melihat distribusi jumlah pengendara pada penduduk Indonesia, sistem tersebut akan sulit digunakan untuk mendeteksi pergerakan pengemudi sepeda motor.

Selain menggunakan model *Deep Learning*, beberapa penelitian menggunakan pembelajaran *Machine Learning* yang lebih sederhana. Model yang paling sering digunakan adalah *Multi-Layer Perceptron* (MLP), dimana sistem yang menggunakan model tersebut memiliki akurasi yang cukup tinggi [4,5,6], namun proses *tuning* yang kurang akurat akan membuat keefisienan model menurun. Model lain yang digunakan adalah model yang menggunakan didasarkan pada *Decision Tree*, dimana model tersebut dapat digabungkan dengan suatu teknik *ensemble* yang disebut *Bagging*, dimana satu model akan menjalani proses pelatihan secara iteratif sehingga menjadi suatu klasifikator yang kuat. Model ini biasa disebut sebagai *XGBoost* [7,8,9]. Proses pelatihan secara iteratif akan memakan *resource* yang tinggi, sehingga model tersebut akan sulit untuk diterapkan pada kondisi *real-time*.

Dengan didasarkan pada alasan dan kelemahan-kelemahan yang terdapat pada sistem-sistem sebelumnya, maka pada penelitian kali ini akan dibuat sistem yang menggunakan pendekatan model *Decision Tree* yang digabung dengan teknik *Bagging* lainnya, yang disebut sebagai *LightGBM* [10,11]. Sistem akan didesain dengan mengutamakan penekanan waktu dan daya komputasi yang dibutuhkan oleh model dalam mendeteksi kondisi pengemudi, dan juga akan diintegrasikan dengan sensor pada *smartphone*, untuk meningkatkan mobilitas dari sistem.

2. Metodologi

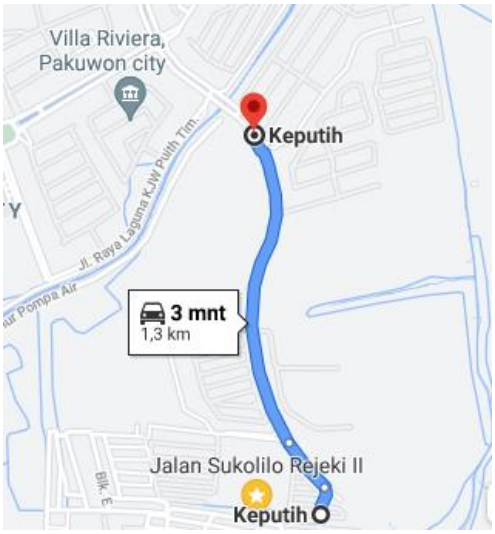
Proses deteksi aktivitas abnormal dari pengemudi sepeda motor dapat dilakukan melalui beberapa tahapan, seperti yang ditampilkan pada Gambar 1. Secara garis besar, sistem dapat dibagi menjadi 3 bagian : Pengumpulan dan *Pre-Processing* data, Proses pelatihan model LightGBM, dan yang terakhir adalah analisa performansi dari model tersebut.



Gambar 1. Desain sistem klasifikasi aktivitas abnormal pada pengemudi sepeda motor

2.1 Pengumpulan Data dan *Pre-Processing*

Dataset yang digunakan pada penelitian kali ini adalah dataset yang dikumpulkan pada penelitian sebelumnya [5], dimana data berasal dari sensor *Accelerometer* dan *Gyroscope* yang terdapat pada *smartphone*. Dataset berasal dari 5 pengemudi motor yang berbeda, dimana masing masing akan melakukan 9 gerakan yang berbeda, dan memiliki kecepatan yang berbeda – beda. Data disampling tiap 100 ms sekali, dengan durasi sampling total adalah 5 s. Dikarenakan menggunakan 2 sensor dengan 3 axis, maka satu gerakan memiliki representasi sebanyak 300 fitur. Keseluruhan data pada dataset adalah 628 data dengan 7 gerakan yang berbeda, yang dibagi menjadi 2 kategori utama. Gambar 2 menunjukkan potongan dari *Google API* sebagai visualisasi dari kondisi medan yang digunakan untuk proses pengumpulan data, dan Tabel 1 menunjukkan tipe gerakan yang akan digunakan pada penelitian kali ini.



Gambar 2. Kondisi rute yang digunakan dalam proses pengumpulan data

Tabel 1. Tipe gerakan yang dideteksi

Nama Gerakan	Kategori
Lurus	**Normal**
Belok Kanan	**Normal**
Belok Kiri	**Normal**
Zig-Zag	**Abnormal**
Pengeraman	**Abnormal**
Percepatan	**Abnormal**
Mengantuk	**Abnormal**

Dikarenakan data yang berasal dari sensor *smartphone* cenderung memiliki komponen *noise* pada pita frekuensi tinggi, maka data tersebut akan difilter dengan menggunakan LPF (*Low Pass Filter*), untuk menekan *noise* tersebut. Tipe filter yang dipilih adalah IIR (*Infinier Impulse Response*), dimana tipe filter ini tidak memiliki tambahan fase *delay*, sehingga tidak menambah waktu komputasi dari sistem.

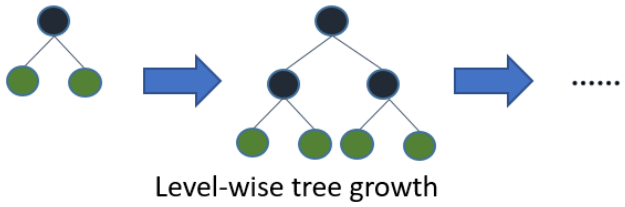
Setelah data difilter, maka proses berikutnya adalah untuk melakukan normalisasi. Tujuan dari proses normalisasi adalah untuk membuat *range* yang digunakan dalam dataset memiliki nilai yang sama. Untuk melakukan proses ini, maka akan digunakan persamaan *Min-Max* akan digunakan. Persamaan 1 menunjukkan rumus yang digunakan untuk normalisasi

$$Xnew = \frac{X_i - \min(X)}{\max(X) - \min(X)} \quad (1)$$

Dikarenakan fitur yang berasal dari dataset asli adalah representasi gerakan tiap 100 ms, dimana jumlah fitur yang dihasilkan adalah 300. Jumlah fitur yang tinggi akan menaikkan pengaruh varians pada proses pelatihan model kecerdasan buatan. Oleh karena itu, pada penelitian ini, akan dilakukan proses *re-sampling*, dimana akan diambil nilai rata-rata dari tiap 500 ms sekali, sehingga jumlah akhir dari fitur yang digunakan adalah 60 fitur. Dengan adanya proses *re-sampling* tersebut, maka varians dari data dapat ditekan, dan juga waktu komputasi akan berkurang nantinya.

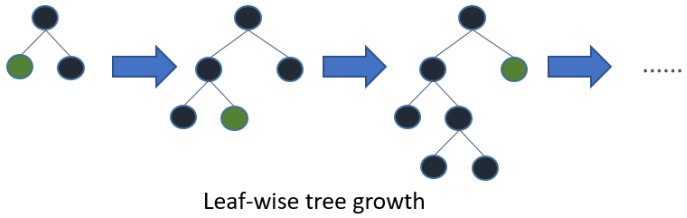
2.2 Proses Pelatihan

Data yang didapatkan dari proses sebelumnya akan menjadi data untuk proses pelatihan dari model pembelajaran mesin (*Machine Learning*) yaitu model LightGBM. LightGBM dipilih karena memiliki keunggulan dalam segi proses komputasi yang jauh lebih cepat dibandingkan dengan model berbasis *Decision Tree* lainnya. Sebagai contoh, pada model XGBoost, menggunakan suatu algoritma pertumbuhan didasarkan pada tingkat level (*level-wise growth*). Model akan terus membelah *node-node* daun pada suatu tingkat yang sama, dimana *splitting gains* dari *node* tersebut bernilai rendah, sehingga dihasilkan suatu sistem yang membuang-buang daya komputasi untuk terus melakukan pembelahan *node*, walaupun proses tersebut sudah tidak dibutuhkan lagi. Gambar 3 menunjukkan diagram proses pembelahan *node* daun pada model XGBoost.



Gambar 3. Teknik pembelahan *node* daun pada model XGBoost

Di sisi lain, LightGBM mengusung teknik pembelahan yang disebut *leaf-wise growth*. Teknik tersebut akan membatasi kedalaman dari model LightGBM, dan mencari *node* daun yang memiliki *splitting gains* paling besar, memecah *node* tersebut, dan meneruskan proses untuk *node* yang baru. Level tambahan tidak akan diperlukan untuk meningkatkan tingkat kemurniaan (*purity*) dari model, yang mencegah model LightGBMm tumbuh terlalu dalam. Sebagai hasilnya, sistem akan jauh lebih sedikit mengonsumsi daya komputasi, dan juga terhindar dari *overfitting*. Gambar 4 menunjukkan teknik pembelahan *node* daun pada LightGBM. Beberapa penelitian yang menggunakan LightGBM [12,13] menunjukkan keunggulan tersebut.



Gambar 4. Teknik pembelahan *node* daun pada model LightGBM

2.3 Analisa Kinerja

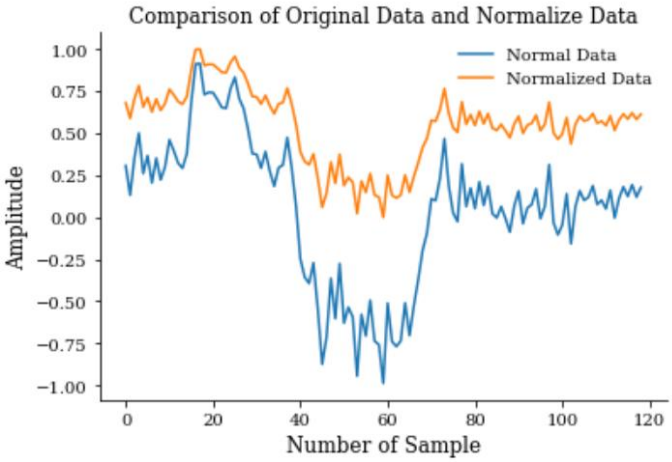
Dari model LightGBM yang telah menjalani proses pelatihan, dan akan dikombinasikan dengan proses *hyperparameter tuning* untuk mendapatkan nilai parameter yang optimal, maka performa dari model akan dianalisa beberapa *metric* yang akan menentukan bagaimana model dapat bekerja dalam implementasi *rea-time*. Dalam analisa performansi yang dilakukan, *metric* yang dipilih adalah *metric* dasar yaitu akurasi, yang dapat dihitung dengan menggunakan persamaan 2.

$$Accuracy = \frac{TP + TN}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \times 100\% \quad (2)$$

Metric kedua yang akan dianalisa adalah terkait dengan waktu komputasi dari model. Untuk menganalisa *metric* tersebut, maka model LightGBM akan diintegrasikan dengan suatu aplikasi *smartphone*, dimana aplikasi tersebut akan membaca data dari sensor *Accelerometer* dan *Gyroscope* yang terintegrasi didalamnya, dan data akan diproses oleh sistem untuk diklasifikasikan oleh model. Waktu komputasi yang dianalisa dimulai dari waktu model mendapat data dan mengklasifikasikan data tersebut.

3. Hasil dan Pembahasan

Pada bagian ini, hasil dari tiap bagian sistem yang telah disebutkan sebelumnya akan dijelaskan. Dimulai dari hasil yang didapat dari pembacaan dan *pre-processing* data sensor, paramater yang krusial dari model LightGBM, dan juga skenario pengujian yang dilakukan untuk melakukan analisa performa dari model. Gambar 5 akan menunjukkan perbandingan antara data sensor yang didapat secara *raw* dan juga data yang telah mengalami normalisasi.



Gambar 5. Perbandingan data *raw* dari sensor *Accelerometer* axis–x dengan hasil normalisasi

Pada Gambar 5, menunjukkan data yang diambil dari axis-x pada sensor *Accelerometer*. Dengan menggunakan proses normalisasi, maka range dari data yang didapat dapat ditekan menjadi range 0-1 untuk semua data dari kedua sensor tersebut. Setelah data dinormalisasi, maka proses berikutnya adalah untuk melakukan *re-sampling* pada data. Proses tersebut digunakan untuk mengurangi jumlah fitur yang terlalu tinggi yang didapat dari data asli.

Setelah data selesai di *re-sampling*, maka akan dilakukan pelatihan model LightGBM. Untuk mendapatkan performa yang optimal, maka diperlukan proses *tuning* untuk beberapa parameter. Pada penelitian kali ini, akan dibahas beberapa parameter yang krusial, yaitu tipe algoritma *bosoting* yang digunakan, dan kombinasi algoritma *regularization* yang digunakan. Tabel 2 menunjukkan parameter dari model yang telah di *tuning*.

Tabel 2. Parameter model LightGBM

Parameter	Nilai
Algoritma *Boosting*	**GBDT**
Estimator	**4000**
Objective	***Multi-Class***
Learning Rate	**0.05**
Regularization Alpha	**0.15**
Regularization Lambda	**0.15**
Metric	***Multi Error***

Parameter *Esitmator* dan *Learning Rate* merupakan parameter yang tidak memiliki nilai yang tetap, dimana nilai yang digunakan datang dari proses *trial-and-error* yang dikombinasikan dengan proses *tuning* untuk mendapat nilai yang optimal dengan memperhatikan *trade-off* antara akurasi model dengan waktu komputasi. *Objective* diset menjadi *multi-class* dikarenakan terdapat 7 kelas berbeda yang akan ditest, dan bentuk label tidak dalam bentuk biner. Oleh karena *objective* tersebut, maka *metric* pada proses pelatihan yang akan diobservasi adalah *multi-error*.

Parameter *Regularization* dibagi menjadi 2 tipe, Alpha dan Lambda. Dalam penelitian kali ini, konsep yang sama dengan penggunaan *ElasticNet* pada model *Logistic Regression*, yang menggunakan baik L1 and L2 penalti [14]. Dalam model berbasis *Decision Tree*, kompleksitas dari model merupakan salah satu aspek yang penting untuk diperhatikan. Model yang terlalu kompleks akan berujung pada kondisi *overfitting*. Untuk menangani hal ini, terdapat beberapa cara yang dapat digunakan, sebagai contoh dengan menggunakan *early stopping* ataupun dengan menggunakan bantuan algoritma *Regularization*. Algoritma tersebut akan diimplementasikan pada *node* daun, dimana *Regularization* akan mengurangi efek dari fitur yang kurang berkontribusi terhadap proses pelatihan model LightGBM, namun tanpa menghilangkan fitur tersebut Persamaan 3 dan 4 adalah persamaan *Regularization* Alpha dan Lambda yang digunakan.

$$Loss = Error\ (Y - \bar{Y}) + \lambda \sum_{I}^{N} |wi| \tag{3}$$

$$Loss = Error\ (Y - \bar{Y}) + \lambda \sum_{I}^{N} |wi * wi| \tag{4}$$

Parameter yang terakhir adalah parameter yang merupakan parameter yang paling penting dari model LightGBM. Algoritma *boosting* akan mementukan bagaimana cara pertumbuhan dari model berbasis *Decision-Tree*. Pada LightGBM terdapat 3 opsi yang dapat dipilih sebagai algoritma *boosting*: GBDT (*Gradient Boosted Decision Tree*), DART (*Dropout meet Multiple Additive Regression Trees*), dan GOSS (*Gradient-based One Side Sampling*).

GBDT adalah algoritma standard yang paling sering digunakan dalam melakukan pelatihan pada model berbasis *Decision Tree*. Algoritma tersebut akan melakukan *training* dimulai satu model dasar secara iteratif, dimana model berikutnya akan menggunakan *error* yang didapat pada proses sebelumnya. Hal ini berbeda dengan model *Random Forest* yang menggunakan beberapa model *Decision Tree* secara independen. Kelemahan dari GBDT adalah bahwa pada iterasi-iterasi terakhir, model akan mengalami *over-specialize* sehingga membuat model hanya memperhitungkan beberapa fitur saja yang memiliki korelasi tinggi dengan hasil proses pelatihan. DART tidak digunakan dalam penelitian kali ini dikarenakan parameter yang digunakan pada DART lebih susah untuk dilakukan *tuning*, dimana hasil *tuning* yang tidak akurat meningkatkan probabilitas *overfitting* pada model. GOSS adalah modifikasi dari GBDT, dimana proses pelatihan difokuskan pada data *training* yang menghasilkan gradien yang bernilai tinggi, sehingga proses pelatihan berjalan jauh lebih cepat, dan kompleksitasan model akan menurun. Dengan GOSS, data *training* yang hanya menghasilkan nilai gradient yang kecil akan dihapus dari proses pelatihan. Walaupun GOSS menghasilkan waktu komputasi dan tingkat akurasi yang cukup tinggi, kelemahan dari GOSS adalah bahwa algoritma tersebut lebih cocok digunakan pada data yang berukuran besar (beberapa penelitian sebelumnya menyarankan ukuran data > 10.000 untuk menggunakan GOSS), dikarenakan ketika diimplementasikan pada data yang berukuran kecil, maka pengurangan data akan berdampak pada proses klasifikasi nantinya. Dikarenakan ukuran data yang digunakan pada penelitian kali ini tidak mencapai volume yang terlalu besar, maka GOSS tidak optimal untuk digunakan pada model LightGBM yang digunakan. Sebagai hasilnya, maka GBDT akan digunakan, dimana algoritma tersebut terkenal menghasilkan model dengan performa akurasi yang cukup tinggi.

Setelah didapatkan model yang telah memiliki parameter yang optimal, maka model akan diuji dengan menggunakan data yang telah disiapkan sebelumnya. Tabel 3 menunjukkan *Confusion Matrix* dari proses klasifikasi padi data *testing*.

Tabel 3. *Confusion Matrix* dari proses klasifikasi model LightGBM

	Lurus	Belok Kiri	Belok Kanan	Zig-Zag	Percepatan	Pengereman	Mengantuk
Lurus	4	0	0	0	0	1	0
Belok Kiri	1	2	0	1	0	0	1
Belok Kanan	2	0	3	0	0	0	0
Zig-Zag	0	0	0	3	0	0	0
Percepatan	0	0	0	0	3	0	0
Pengereman	0	0	0	0	0	3	0
Mengantuk	0	0	0	0	0	0	3

Berdasarkan Tabel 3. Model LightGBM menghasilkan hasil yang sangat akurat untuk gerakan yang masuk ke dalam kategori Abnormal. Untuk kategori tersebut, model yang digunakan menunjukkan tingkat akurasi 100%. Di sisi lain, ketika digunakan untuk mendeteksi kondisi normal, maka keakuratan model menurun, dikarenakan beberapa gerakan yang sifatnya kurang dari waktu *sampling*. Untuk gerakan-gerakan seperti Belok Kiri dan Belok Kanan, tidak semua kondisi dalam lalu lintas membuat pengemudi dapat melakukan gerakan tersebut dalam jangka waktu 5 detik. Oleh karena durasi gerakan yang singkat, sensor menangkap campuran beberapa gerakan dalam satu *window sampling*. Total akurasi untuk keseluruhan dataset adalah 82.8%.

Tabel 4. Akurasi dari proses klasifikasi model LightGBM pada data *real-time*

Gerakan	**Kecepatan 20 – 30 Km/ Jam**	**Kecepatan 30 – 40 Km/ Jam**
Lurus	67%	60%
Belok Kanan	75%	75%
Belok Kiri	37.5%	37.5%
Zig-Zag	100%	100%
Percepatan	70%	100%
Pengereman	50%	40%
Mengantuk	67%	67%
Rata-Rata	66.6%	68.5%

Tabel 4 menunjukkan performa model ketika model diintegrasikan kedalam suatu aplikasi *smartphone* untuk deteksi secara *real-time*. Dapat disimpulkan bahwa model yang dihasilkan bekerja secara efektif pada kecepatan 30-40 Km/ Jam. Untuk meningkatkan akurasi model, data

yang berukuran lebih besar yang memiliki varian kecepatan diperlukan untuk meningkatkan performa dari model.

Tabel 5. Waktu komputasi dari model LightGBM pada deteksi *real-time*

Iterasi	Waktu Komputasi (mili detik)
1	10
2	10
3	12
4	12
5	14
6	10
7	9
8	11
9	10
10	10
Rata-Rata	10.8

Tabel 5 menunjukkan waktu komputasi model yang diobservasi pada 10 iterasi yang berbeda. Untuk menghasilkan suatu hasil klasifikasi, model LightGBM yang digunakan hanya membutuhkan waktu rata-rata 10 mili detik. Artinya bahwa model ini bekerja lebih cepat 5 kali lipat dibandingkan dengan model lama yang menggunakan pendekatan *Multi-Layer Perceptron* [5], dan bekerja lebih cepat 100 kali lipat dibanding sistem yang menggunakan pendekatan *Deep Learning* yang memiliki waktu komputasi rata-rata 1 detik. Hal ini membuktikan bahwa model yang digunakan memiliki keunggulan yang sangat tinggi dalam segi waktu dan konsumsi daya komputasi.

4. Kesimpulan

Dari hasil yang didapat dan dijelaskan diatas, dapat disimpulak beberapa hal sebagai berikut:

1. Sistem bekerja dengan lebih baik pada kecepatan 30 – 40 Km/Jam, dibandingkan dengan kecepatan 20 – 30 Km/Jam, dengan akurasi 68.5%. Untuk meningkatkan performa model, dataset yang lebih besar diperlukan, Ukuran data yang lebih besar akan memungkinkan penggunaan algoritma GOSS untuk meningkatkan performa model.
2. Sistem yang dihasilkan memiliki waktu komputasi rata – rata 10 mili detik, dimana artinya sistem bekerja 5 kali lipat lebih cepat dibanding pendekatan MLP, dan hampir 100 kali dibanding pendekatan *Deep Learning*. Hal ini membuat sistem pada penelitian kali ini lebih efisien ketika digunakan dalam pendeteksi *real-time* dengan menggunakan *device* yang memiliki daya komputasi yang terbatas (sebagai contoh : *smartphone*).

Daftar Pustaka

[1] Jusuf, A., Nurprasetio, I. P., & Prihutama, A. "*Macro data analysis of traffic accidents in Indonesia*". Journal of Engineering and Technological Sciences. 2017.

[2] Eraqi, H. M., Abouelnaga, Y., Saad, M. H., & Moustafa, M. N. "*Driver distraction identification with an ensemble of convolutional neural networks*". Journal of Advanced Transportation.2019.

[3] Shahverdy, M., Fathy, M., Berangi, R., & Sabokrou, M. "*Driver behavior detection and classification using deep convolutional neural networks*". Expert Systems with Applications. 2020.

[4] Brombacher, P., Masino, J., Frey, M., & Gauterin, F. "*Driving event detection and driving style classification using artificial neural networks*". Proceedings of the IEEE International Conference on Industrial Technology. 2017.

[5] Nuswantoro, F. M., Sudarsono, A., & Santoso, T. B. "*Abnormal driving detection based on accelerometer and gyroscope sensor on smartphone using artificial neural network (ann) algorithm*". IES 2020 - International Electronics Symposium: The Role of Autonomous and Intelligent Systems for Human Life and Comfort. 2020.

[6] Matousek, M., El-Zohairy, M., Al-Momani, A., Kargl, F., & Bosch, C. "*Detecting anomalous driving behavior using neural networks*". IEEE Intelligent Vehicles Symposium, Proceedings. 2019.

[7] Chen, T., & Guestrin, C. "*XGBoost : Reliable Large-scale Tree Boosting System*". ArXiv. 2016.

[8] Shi, X., Wong, Y. D., Li, M. Z. F., Palanisamy, C., & Chai, C. "*A feature learning approach based on XGBoost for driving assessment and risk prediction*". Accident Analysis and Prevention. 2019.

[9] Lu, Y., Fu, X., Guo, E., & Tang, F. "*XGBoost Algorithm-Based Monitoring Model for Urban Driving Stress: Combining Driving Behaviour, Driving Environment, and Route Familiarity*". IEEE Access. 2021.

[10] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. "*LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems*". 2017

[11] Zeng, H., Yang, C., Zhang, H., Wu, Z., Zhang, J., Dai, G., Babiloni, F., Kong, W., & Chuang, L. "*A LightGBM-Based EEG Analysis Method for Driver Mental States Classification*". Computational Intelligence and Neuroscience. 2019.

[12] Gao, X., Luo, H., Wang, Q., Zhao, F., Ye, L., & Zhang, Y. "*A human activity recognition algorithm based on stacking denoising autoencoder and lightGBM*". Sensors (Switzerland). 2019.

[13] Choi, S., & Hur, J. "*An ensemble learner-based bagging model using past output data for photovoltaic forecasting*". Energies. 2020.

[14] Demir-Kavuk, O., Kamada, M., Akutsu, T., & Knapp, E. W. "*Prediction using step-wise L1, L2 regularization and feature selection for small data sets with large number of features*". BMC Bioinformatics. 2011.