

# Detection of traffic congestion based on twitter using convolutional neural network model

Rifqi Ramadhani Almassar, Abba Suganda Girsang

Department of Computer Science, Binus Graduate Program-Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia

## Article Info

### Article history:

Received Nov 8, 2021

Revised May 12, 2022

Accepted Jun 10, 2022

### Keywords:

Convolutional neural network

Text classification

Traffic congestion

Twitter

Word embeddings

## ABSTRACT

Microblogging is a form of communication between users to socialize by describing the state of events in real-time. Twitter is a platform for microblogging. Indonesia is one of the countries with the largest Twitter users, people can share information about traffic jams. This research aims to detect traffic jams by extracting tweets in the form of vectors and then inserting them into the Convolution neural network (CNN) model and getting the best model from CNN+Word2Vec, CNN+FastText, and support vector machine (SVM). Data retrieval was conducted using the Rapidminer application. Then, the context of the tweets was checked so that there were 2777 data consisting of 1426 congestion road data and 1351 smooth road data. The data was taken from certain coordinate points in around Jakarta, Indonesia. Then, preprocessing and changes to vector form were carried out using the Word2Vec and FastText methods, then inserted into the CNN model. The results of CNN+Word2Vec and CNN+FastText were compared to the SVM method. The evaluation was done manually using the actual traffic conditions. The highest result obtained using test data by the CNN+FastText method are 86.33% while CNN+Word2Vec is 85.79% and SVM is 67.62%.

This is an open access article under the [CC BY-SA](#) license.



## Corresponding Author:

Rifqi Ramadhani Almassar

Department of Computer Science, Binus Graduate Program-Master of Computer Science, Bina Nusantara University

Jakarta, Indonesia, 11480

Email: rifqi.almassar@binus.ac.id

## 1. INTRODUCTION

Information technology is growing rapidly including social media. Nowadays, it easy for people to access social media in order to interact or seek entertainment through social media. Twitter is one of social media for microblogging. Microblogging is a form of communication among internet users to socialize by describing the state of events through the web [1]. Users can send and read messages on Twitter which are known as a tweet, but twitter limits the characters in the message.

In October 2020, Twitter users from Indonesia reached 13.2 million, making Indonesia the 5th largest after Turkey [2]. The data shows that Twitter is widely used by people in Indonesia. Twitter users can share information and connect with other users in real-time. Users can also provide opinions about what is happening in real-time, for example providing information and the state of the traffic they are passing.

Indonesia is one of the developing countries that have such problems as high population growth, inadequate infrastructure, and traffic congestion. High population growth results in high levels of urbanization and transportation needs, causing traffic jams. Congestion can cause socio-economic problems. Tweet data can be used to analyze congestion conditions on a road. The main advantage of congestion analysis using tweet data is that real-time data is obtained directly from users based on tweet time so that they

can view traffic history according to the tweet date and if applied to a cellphone application, it will save battery because it will not run continuously like Global positioning system (GPS).

In [3], image with foreground estimation and cascade classifier were used to detect congestion, but this research used text from Twitter to detect traffic congestion. Based on research [4], doing preprocessing by eliminating URLs, or special words before classification, while in [5] hashtag symbols were also removed, and in [6] deleting unrelated words. Furthermore, to be included in a model according to research by [7], feature representation can be carried out using the Bag-of-words (BOW) technique to be numerical, while in [8] feature representation is carried out using the Term frequency-inverse document (TF-IDF) method so that it becomes a matrix that can be implemented in the Long short-term memory (LSTM) method. In this study, we use the Word2Vec method, which combines Skip-gram and Continuous bag-of-Words (CBOW) and FastText. It is the development of Word2Vec which has the advantage of being able to process words that are not in the vocabulary. Word2Vec and FastText can be used in conjunction with the Convolutional neural network (CNN) method to do the classification.

Research on tweet-based traffic jam detection has been widely conducted using various methods such as taking words from each tweet and then classifying the words with the Support vector machine (SVM) model [9]. Alhumoud [10] obtained Twitter data, entered the data into AsterixDb and then classified it using Spark. According to research [11], preprocessing stemming and tokenization were applied to the data. Dabiri and Heaslip [12] carried out mapping tweets into vectors to measure the relationship between words and then enter them into the CNN and Recurrent neural network (RNN) models to get traffic incidents. CNN is used to classify by extracting features and word representation of each tweet. By using the Word2Vec model to distribute words and then enter them into CNN, we can get better extraction results [13]. Currently, the best results in tweet-based congestion detection research are obtained using the SVM method and the results reach 96.24% [9]. In [12] research predicting traffic incidents, it can reach 98.6% by using the CNN method. Therefore, this study predicts traffic congestion using Twitter data by using the CNN method. So, this research aims to congestion detection by classifying tweets and considering the frequency of tweets, by going through the preprocessing process, extracting features through Word2Vec and FastText, and then entering them into the CNN model through the Tensorflow framework with Python programming language and then comparing the two with data testing.

## 2. RELATED WORKS

Several studies have focused on detecting traffic using tweet data with various approaches. D'Andrea *et al.* do classified tweets or in this study called Status update message (SUM) to detect traffic using several methods [14]. The authors collected Italian-language tweets and then conducted preprocessing by tokenization and cleansing. The last preprocessing performed feature representation with TF-IDF in numerical vector form. The study obtained SVM with the highest accuracy of 95.75%.

In research conducted by Yang *et al.* [1], performs traffic detection with deleted tweets if they are not related to traffic and performs symbol encoding. Based on the results of the analysis on each tweet, the authors argue that there are elements of statements that determine the traffic situation, and there are hashtags that do not always have meaning. So, it is possible to conduct traffic analysis prediction research based on tweets.

Gu *et al.* [15] in detecting traffic after collecting Twitter data then removed symbols and synonyms were extracted from the WordNet database and the authors performed tokenization. Tweet classification was done using the supervised Latent dirichlet allocation (sLDA) model and the Semi-naïve bayes method. The study obtained accuracy of 90.5%.

Zulfikar and Suharjito [9] used Twitter data labelled using the X-Means Clustering algorithm. The authors used the TF-IDF method and performed cross-validation. Furthermore, the authors used the k-NN and Naïve bayes methods, and then compared the results with the SVM method. The results of the performance evaluation using the confusion matrix sigmoid SVM method were 96.24%.

Dabiri and Heaslip built a traffic incident detection model based on Twitter using deep learning architecture [12]. Then the classification was carried out in 3 ways, namely only CNN, only LSTM, and a combination of CNN and LSTM. The results of this study indicate that using only CNN and converting vectors into 2 classes with Word2Vec or FastText models and the highest accuracy, which is 98.6%.

Alhumoud explained that an Intelligent transportation system (ITS) is a solution for monitoring traffic and travel efficiently [10]. The data stored using AsterixDB is as much as possible. Then, the data is filtered, and data is obtained which is the result of research. Using the spark data processing engine connected to AsterixDB provides 81%, 89%, and 92% precision for accidents, weather conditions, and road incidents.

Wongcharoen and Senivongse [16] predicted traffic jams in Thailand using data from several Twitter accounts. The author labelled the data as high (H), medium (M), and low (L). Next, they trained the Decision tree (DT) using C4.5 with the Weka application. The results of the study with evaluation using 10-

fold cross-validation obtained precision 0.89, recall 0.898, F-measure 0.892, and receiver operating characteristic (ROC) Area 0.894.

Gong *et al.* [17] explained that tweets often include geospatial data such as the user's profile, latitude/longitude, and time of the tweet. Next, clustering is done using the Density-based spatial clustering of applications with noise (DBSCAN) algorithm and the authors also does clustering using the ELKI Data mining framework to get two distances between tweets. The clustering results are then stored in Apache CouchDB and visualized on a map using the website.

Septianto *et al.* [18] collected tweet data from Twitter then did preprocessing. Then, Naïve bayes was done to determine the label. Then, the name of the place was taken by dividing between the words "direction" and "towards" so that traffic is taken from the 2 place names. The results are then visualized on Google Maps and repeated every 5 minutes. The results of this study obtained an accuracy of 61.66%.

Subhan *et al.* [4] collected tweet data from Twitter application programming interface (API) then change the words in the tweet into standard words via kateglo.com. Next, perform a semantic analysis using the Generative lexicon method, the results obtained are the location information from and the destination was separated by the words "menuju", "ke", or "sampai". The results of this study were visualized with Google Maps to validate the results the authors compared with the actual conditions.

Ankit and Saleena conducted a sentiment analysis of tweets whether they contained positive or negative sentiments [7]. The authors performed feature representation using the BOW technique, which represents tweets as numeric. Classification uses several techniques, namely Naïve bayes, random forest (RF), SVM, Logistic regression (LR). Then, the ensemble classifier is carried out by combining the classification techniques that have been carried out to improve accuracy, using several data sets, the highest accuracy results are obtained using the ensemble classifier as much as 85.83%.

Ansari *et al.* [8] conducted a sentiment analysis of the 2019 presidential election in India. The authors performed feature extraction based on Term-Frequency to convert data into vectors. Then, it classified machine learning with the SVM algorithm, DT, LR, RF, and LSTM. The highest result was obtained by LSTM with 74% accuracy.

Khan *et al.* introduced Twitter opinion mining (TOM) framework for classified positive, negative, and neutral sentiments [19]. The Authors performed preprocessing by detecting slang words that will be interpreted based on the WordNet, Netlingo, and SMS dictionary libraries. The classification algorithm in the framework used 3 basics, namely Enhanced emoticon classifier (EEC), Improved Polarity Classifier (IPC), and SentiWordNet classifier (SWNC). Based on the results of this study using the TOM framework, an average accuracy of 85.7% was obtained with 85.3% precision and 82.2% recall.

Ruz *et al.* [6] conducted sentiment analysis during natural disasters or political transitions. Using the Twitter dataset, the first dataset was on the Chile earthquake and the second dataset was taken during the Catalan independence referendum. The authors performed feature representation using the BOW technique, to overcome unbalanced sentiment analysis, then apply the Synthetic minority over-sampling technique (SMOTE). Based on this study, the highest results were found in the first dataset is SVM with an accuracy of 81.2%, and the second dataset is RF with an accuracy of 85.8%.

Lal *et al.* performed tweet data analysis of tweets data to identify criminal events that require police attention [5]. Before being analyzed the data was preprocessed by dividing the word into several segments in 1 sentence, then the hashtag word was deleted. The authors perform the TF-IDF conversion and by using the Waikato environment for knowledge analysis (WEKA) application. The results of the classification using the RF algorithm get the highest accuracy, which is 98.1%.

Yao and Qian [20] aims to check traffic before 5 a.m. using the previous day's traffic predictions taken from Twitter. In that study, the tweet data was augmented to reduce noise data based on the tweet clock and geocode. After that, the data is extracted to get the traffic the next day. By using the tweet2traffic clustering model for congestion, the accuracy is 79%.

Ahmed *et al.* explained congestion and traffic information can be identified using the Advanced traveler information system (ATIS), but these tools are expensive and cannot be installed throughout the city [11]. The authors performed a Geo-filter to get the location of the tweet in question. The data is then converted into vectors using TF-IDF and clustering using K-Means, then classified using the SVM model to get the highest precision of 82%.

Alomari *et al.* [21] proposed congestion detection and incident detection using Arabic-language tweets. By using systems, applications, and products in data processing high-performance analytic appliance (SAP HANA), which is connected to the Tweet API, then the tweet is processed first with the help of SAP HANA through the tokenization, normalization, and entity extraction stages. Furthermore, sentiment analysis was carried out using SAP HANA and assisted by SAP Lumira to produce visualizations.

Zhang *et al.* detect traffic incidents using the deep belief network (DBN) and LSTM methods [22]. So, the authors did preprocess by labeling and extracting each word manually through tokenization and

stemming. Next, tokenize using DBN and LSTM and then compare it with artificial neural network (ANN) and sLDA. The final result shows that DBN has the best accuracy of 93%.

Essien *et al.* [23] make traffic predictions using Twitter data, traffic, and weather. For Twitter data is taken using the Twitter API based on the @OfficialTfGM and @WazeTrafficMAN accounts, weather data is taken from the Center for Atmospheric Studies (CAS) and traffic data is taken using detectors. The 3 datasets are combined into one and then entered into the Bidirectional long short-term memory (BiLSTM) model using k-fold cross-validation. The results of the evaluation using the (Root mean squared error) RMSE get 6.85%.

Ali *et al.* [24] detected incidents and traffic conditions using ontologies latent Dirichlet allocation (OLDA), and BiLSTM. Data is retrieved from Twitter and Facebook using the API. Next, do a positive, neutral, or negative sentiment analysis to find out the traffic conditions. Then, the data is embedded using FastText and Word2Vec and finally, training is carried out using BiLSTM with softmax regression to classify conditions and predict polarity. The results in this study achieved an accuracy of 97%.

Alomari *et al.* [25] in his research, he created a tool to detect traffic-related events using Twitter data called Iktishaf. It starts by retrieving data via the Twitter API and putting it into MongoDB. The authors conducted TF-IDF to determine the important words in the dataset. Using spark machine learning (ML), tweets classified relevant tweets using Logistics regression, SVM, and Naïve Bayes and delete irrelevant tweets. The results showed that the highest accuracy obtained using SVM reached 90%.

Utari *et al.* [26] research on natural language processing used data from Twitter accounts that inform road and traffic conditions. Using data from Twitter, each tweet was separated by token and each word in the token is analyzed to generate a syntactic structure. The first approach uses anatomy as a determinant of word classification, then classification approaches from other words that are close to unknown words and finally used a notional or contextual approach. The classification results are then entered into a web page. The accuracy in this study reached 88.57%.

Salazar-Carillo *et al.* [27] in detected congestion, taking data from Twitter and then standardizing the words that have been separated based on the n-grams, then identifying traffic flows using the Support vector regression (SVR) library from the results of the accuracy reaching 95.5%. Next, perform the geocoding process to get the visualization predicted by the model. From these results, in addition to congestion, information is also obtained related to events that occur on the road.

### 3. PROPOSED METHOD

Traffic congestion in Indonesia is a problem that often occurs. Many Indonesian people are stuck in traffic jams. So, road users need to know the traffic conditions to avoid congestion. With the development of technology, Twitter is one of the mostly used social media for Indonesian people just looking for entertainment or interacting and one of microblogging platform to socialize by describing the state of events in real-time [1]. During traffic jams, many people provide information or comments about the roads they pass using Twitter.

Indonesia is included in the list of countries with the biggest users of Twitter after Turkey [2]. With the number of Twitter users, the tweet data at the time of congestion can be processed to show the state of the road at the time the tweet was made. Research on congestion detection using data from Twitter has been done before. Twitter has an API that can be used by application developers to retrieve tweet information, user profiles, messages, and other information through an endpoint [12].

In conducting research, [9] took data from the Twitter API and then processed it by taking weights from the text using TF-IDF and then classified it using the SVM [12] research predicted traffic incidents using the CNN to get high accuracy. In this study, we compared the accuracy results between CNN+Word2Vec, CNN+FastText, and SVM. This study processed tweet data on Twitter related to traffic so that it can be used as information on traffic jams on a road using the CNN method.

Based on Figure 1, the first step is to collect tweet data with the Indonesian language keywords "macet" which means congestion road, and "lancar" which means smooth roads from Twitter using the Rapidminer application. The results of the tweets data are then labeled manually according to the traffic jam on Google Maps. Then, the data that is out of context was removed by checking one by one using Microsoft Excel.

Then, preprocessing was done because according to [28] a good preprocessing is needed to get effective and efficient data. In this research, preprocessing was done by encoding labels so that the label "macet" becomes 0 and "lancar" becomes 1. After that, duplicate data was removed so that each tweet data is not the same. Next, cleansing was performed by removing symbols or characters that have no meaning and are not needed in the classification. Then, transform case was performed, which is changing all letters to lowercase letters to facilitate the classification process. After doing the transforming case, then the word was changed to synonyms in Indonesian "macet" to "macet" "padat" "mandek" "tertahan" randomly and the word "lancar" to "lancar" "mulus" "lenggang" randomly to reduce discriminatory features. After that, tokenization was done to separate the words according to the order in the tweet. Each token was done with a stopword in Indonesian to remove words that have no meaning so that the word will not be considered in the

classification. The word on each token is also done stemming so that it changes the word into the core word. Then because CNN is a classification method that requires input in the form of a vector from an image but in this research, it is text data, it needs to be converted into a vector form. In this process, we will represent tweets into vector form using Word2Vec because according to research [29] accuracy with pre-trained vectors gets higher accuracy so words that have the same meaning will get almost the same weight. Word2Vec is an algorithm model that represents words into a vector using a neural network framework [30].

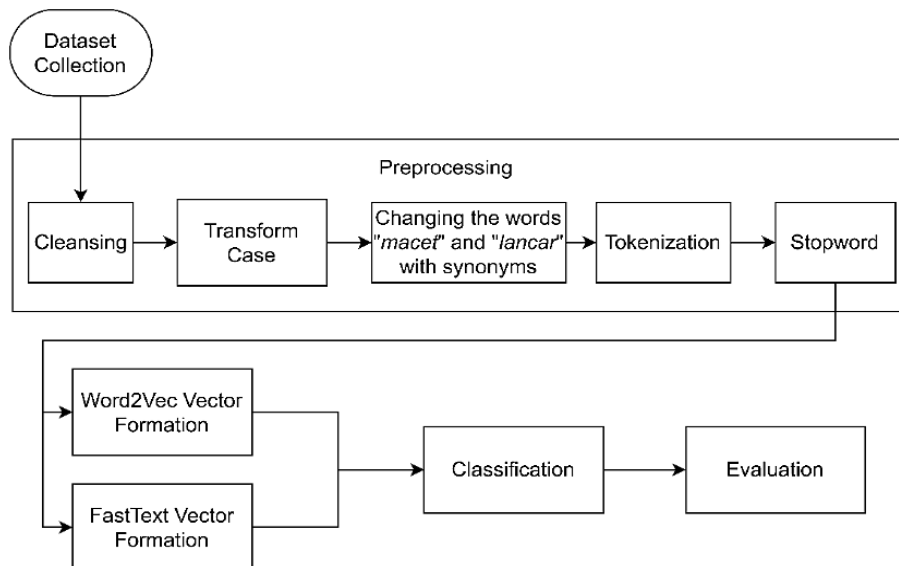


Figure 1. Research stages

This research also used FastText to convert words into vectors for comparison. FastText has an n-gram parameter, which is a text breakdown of n words so that words that are not in the corpus can be used as vectors. In this study, several variations of parameter values were tested and the results are not too different. The Word2Vec and FastText parameters used in this study can be seen in Table 1. Furthermore, in the classification stage, the CNN architecture model used will combine the results from the Pooling Layer with kernel sizes 2, 3, and 4 so that it looks like in Figure 2.

In [8], the classification of tweets is divided into 2, namely “Macet” and “Bukan-Macet” so that in this study the classification is also carried out in 2 classes “macet” class containing tweets about traffic jams based on point coordinates and “lancar” class contains tweets about smooth traffic based on point coordinates. CNN is made with several layers so that it gets a classification from tweets. The description of each parameter on CNN can be seen in Table 2 and Table 3 is a list of layers and their CNN settings in this study.

In text classification using CNN, the results of 1-dimensional vectors that have been carried out using Word2Vec and FastText are then embedded so that they are divided into several channels and stored in the first layer in the vector, then entered into a convolutional layer with kernels 2, 3, and 4. Each convolutional layer result is continued into the pooling layer. The pooling layer results from each kernel are then combined and entered into a fully connected layer consisting of dense, dropout, and dense again, then the activation function is carried out using sigmoid.

Table 1. Word2Vec and FastText parameter

Parameter	Description	Value	
		Word2Vec	FastText
Size	The sum of the dimensions of the vector	100	100
Negative	Number of noise words	5	5
Window	Distance measure from the data context	5	5
Min_count	Minimum number of words available	1	1
Workers	The number of threads used to train the model	8	8
Alpha	Learning rate value	0.065	0.065
Min_alpha	Learning rate drop	0.065	0.065
Word_ngrams	The maximum length of the word ngram	None	3

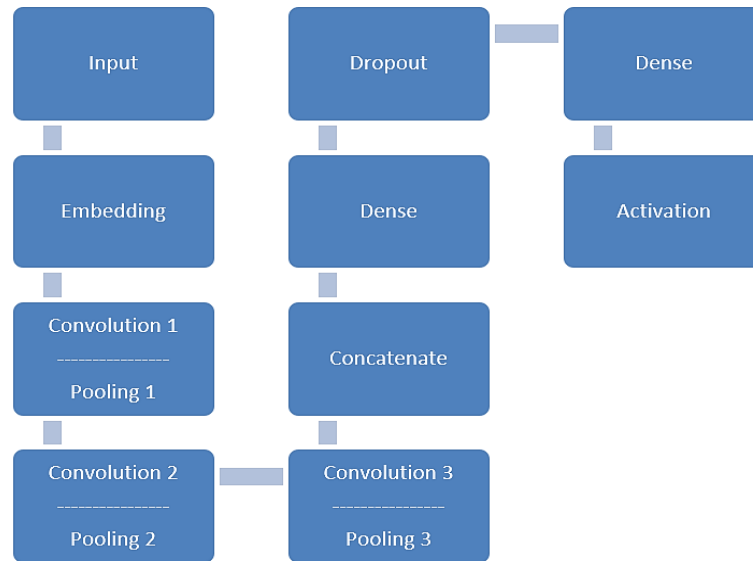


Figure 2. CNN model architecture

Table 2. CNN parameter description

Parameter	Description
input_dim	Number of vocabulary
output_dim	Embedding dimensions
input_length	Input sequence length
filters	Output dimension of convolution layer
padding	Vectors added on each side of the matrix
activation	Activation function
kernel_size	Convolution window length in each step
strides	Convolution step length
units	Output dimension
rate	Part of the input unit to be omitted

Table 3. CNN layer list

Layer	Output	Parameter	Setting
Input Layer	(None, 43)	0	
Embedding	(None, 43, 200)	20000000	input_dim = 100000; output_dim = 200; input_length = 43
Convolution 1	(None, 42, 100)	40100	filters = 100; kernel_size = 2; padding = "valid"; activation = "relu"; strides = 1
Convolution 2	(None, 41, 100)	60100	filters = 100; kernel_size = 3; padding = "valid"; activation = "relu"; strides = 1
Convolution 3	(None, 40, 100)	80100	filters = 100; kernel_size = 4 padding = "valid"; activation = "relu"; strides = 1
Average Pooling 1D 1	(None, 100)	0	
Average Pooling 1D 2	(None, 100)	0	
Average Pooling 1D 3	(None, 100)	0	
Concatenate	(None, 300)	0	
Dense	(None, 256)	77056	units = 128; activation = "relu"
Dropout	(None, 256)	0	rate = 0.1
Dense	(None, 1)	257	units = 1
Activation	(None, 1)	0	activation = "sigmoid"

Based on Keras library documentation on Python programming, if the result is less than 5, then it returns a sigmoid value close to zero and more than 5 then the sigmoid result is close to 1. If the sigmoid result is close to zero then the tweet is in the “macet” which means congestion category, while if the result is close to 1 then the tweet is in the “lancar” which means smooth category. The results of each tweet will be calculated by the number of congestion and smooth, the most results are the categories of traffic at these coordinates.

The dataset is divided into three, namely 60% training, 20% test, and 20% validation. Training is used to train the model so that it can increase its accuracy. For each epoch iteration, validation was carried

out to determine the increase in the accuracy of the model, and the test is used to perform testing when the model has completed training.

To calculate the performance of this study, the confusion matrix is used and the result from CNN+Word2Vec, CNN+FastText, and SVM are compared. The confusion matrix is in the form of a comparison table which contains the number of test data and the number of prediction results using the model that has been made true or not. In the confusion matrix, there are several values, namely True positive (TP) which is the number of data jams that are predicted to be correct, True negative (TN) which is the amount of current data that is predicted to be correct, False positive (FP) which is the amount of current data that is predicted to be false and False negative (FN) which is the number of data crashes that are predicted to be incorrect. This evaluation was divided into 2, namely evaluation of tweet data with testing data and evaluation of actual congestion manually. In the evaluation of tweet data, each data will be predicted using data testing and then entered into the confusion matrix. Meanwhile, the evaluation of actual congestion will be done manually by determining the coordinates and hours to be used for 30 different locations or hours, then taking tweets with the RapidMiner application according to the coordinates and hours. For each data in 1 data set, congestion classification is carried out and the classification results in 1 set will be taken from the most classifications. Then the results in the 1 set classification will be compared with traffic conditions on Google Maps. The results of the actual congestion evaluation manually will also be entered into the confusion matrix.

## 4. RESULTS AND DISCUSSION

### 4.1. Data collection

The data used in this study is tweet data taken using the Rapidminer application with the keywords “*macet*” and “*lancar*” with several coordinate points around Jakarta, Indonesia. The data that was successfully retrieved were 4087 tweets during the months of February 2021-June 2021. Then the data is checked manually by comparing the labels on Google Maps according to the hours and coordinates, then looking at the actual situation according to the keywords “*macet*” or “*lancar*”. In the dataset, it can contain congestion tweet data category but in reality, it is smooth and vice versa. The results were manually checked again to find out the context of the tweets related to traffic or not so that checking the dataset became 2777 tweets consisting of 1426 data congestion or labeled “*macet*” and 1351 data smooth or labeled “*lancar*”. An example of a dataset can be seen in Table 4.

Table 4. Dataset examples

Created-At	From-User	From-User-Id	Text	Retweet-Count	Label
08/02/2021 16:50	Maria_	60450573	"RT @SonoraFM92: 16.40 #LalinSONORA-Jembatan Dua arah Pluit.... Macet <a href="https://t.co/nSNtTslLh8">https://t.co/nSNtTslLh8</a> <a href="https://t.co/0sCgHy09mD">https://t.co/0sCgHy09mD</a> "	2	Macet
08/02/2021 16:34	Ahmad Haris	95354503	<i>jalan an mulai underpass soleh iskandar bogor, 2 minggu ini macet mulu yang arah ke cibinong. problemnya cuma jalan berlubang pas jembatan.</i>	0	Macet
10/03/2021 07:33	Radio Sonora Jakarta	180696019	"07.00 Situasi arus lalu lintas terkini di Jl. RS Fatmawati menuju Cipete Raya/Blok M, dan sebaliknya menuju TB Simatupang/Lebak Bulus terpantau lancar. <a href="https://t.co/jc6K9D5FMF">https://t.co/jc6K9D5FMF</a> @TMCPOdaMetro"	1	Lancar
10/03/2021 07:37	PT. JASAMARGA	540625251	07.37 WIB #Tol_JLJ Ulujami-Pd Ranji-Serpong LANCAR.; Serpong-Pd Ranji-Ulujami LANCAR.	0	Lancar

### 4.2. Preprocessing

Furthermore, to make it easier to classify the label “*macet*” is changed to 0 and “*lancar*” is changed to 1. The next preprocessing in this process includes deleted unused columns, cleansing, transform case, tokenization, stopword and stemming. This stage is carried out with the literature and NLTK libraries.

**Delete unused columns.** The deleted columns are Created-At, From-user, From-user-id, To-user, To-user-id, language, source, Geo-location-latitude, Geo-location-longitude, Retweet-count so that leaves a column Text and Label. **Cleansing.** Cleansing is the process of removing symbols or characters that have no meaning and are not needed in the classification. Example: *RT SonoraFM92 16.40 LalinSONORA Jembatan Dua arah Pluit Macet.* **Transform Case.** Transform Case is the process of changing all letters to lowercase to facilitate the classification process. Example: *rt sonorafm92 16.40 lalinsonora jembatan dua arah pluit macet.* **Changing**

the words “*macet*” and “*lancar*”. Changed into synonyms in Indonesian, the word “*macet*” becomes “*macet*”/”*padat*”/”*mandek*”/”*tertahan*” at random, and the word “*lancar*” becomes “*lancar*”/”*mulus*”/”*lenggang*” randomly to reduce discriminatory features. Example: *rt sonorafm92 16.40 lalinsonora jembatan dua arah pluit padat*. **Tokenization.** Tokenization is the process of separating words according to the order in the tweet. Example: [*rt, sonorafm92, 1640, lalinsonora, jembatan, dua, arah, pluit, padat*]. **Stopword dan Stemming.** Stopword is the process of removing words that have no meaning so that the word will not be considered in the classification and stemming is the process of changing words into their basic word form. In this study, the stopwords and stemming processes become 1 process and the list of stopwords used is taken from Tala Stopword. Example: *rt sonorafm92 1640 lalinsonora jembatan arah pluit padat*.

#### 4.3. Convert to vector Word2Vec

In this process, the data has gone through the preprocessing process so that the data can then be used for word embedding processes. The word embedding process in this study uses the Gensim library. In this process, input data that has been preprocessed in the form of tokenized text is used and will produce a model. Each tweet in the dataset is taken with a different word using the TaggedDocument library. Then each word is entered into Word2Vec.

Word2Vec uses a neural network to get the initial weight values, consisting of 2 architectures are skip-gram and CBOW. In this study, Word2Vec was trained with the CBOW model with 50 epochs and then trained with the skip-gram model with 50 epochs. The results of the model with the skip-gram architecture will be combined with the CBOW model. Table 5 is an example of the final Word2Vec result.

Table 5. Word2Vec final result example

Word	Vector
<i>jalan</i>	-4.039771, 1.4529599, -0.72466475, 0.14262459, -6.820671, -3.7326174, 0.60550255, -5.2219214, 1.0444719, -3.082483, ..., 1.0215762, 0.11307827, -0.6540826, 0.07509925, 1.8175708, 0.87743133, -0.5211345, 1.354927, -0.08151489, 0.43929535, ...
<i>padat</i>	-2.28622794e+00, 7.76517034e-01, -1.54419556e-01, 8.57797921e-01, -3.03995442e+00, -4.07718992e+00, 3.46807949e-02, ..., -7.54858077e-01, 7.09322989e-01, 1.37110937e+00, 5.55417128e-02, -1.02720189e+00, -2.23155960e-01, 5.28727114e-01, ...
<i>cakung</i>	1.4838842, 0.13154435, -0.10948927, -1.2377591, -0.04943107, -0.55506617, 0.5633306, -0.5293238, 0.52347064, 0.5164963, ...
<i>rorotan</i>	-0.5595496, 0.45256287, -0.8794082, 0.5641887, 0.1843323, -0.3603339, 0.2183291, 0.17377803, 0.01006237, -0.42547587, 0.15067908, ...
<i>lenggang</i>	

#### 4.4. Convert to vector FastText

In addition to using Word2Vec vector formation also uses FastText which is a development of Word2Vec by dividing words into n-grams. This process uses the Gensim library, by entering data in the form of text that has passed tokenization and then goes through training and produces a model. By using TaggedDocument the words entered in the training will be different as in the vector formation stage using Word2Vec. FastText also uses a neural network consisting of 2 architectures are skip-gram and CBOW. Next, training the FastText with the CBOW model with 50 epochs and also training with the skip-gram model with 50 epochs. The results of the model with the skip-gram architecture are combined with the CBOW model as shown in Table 6.

Table 6. FastText final result example

Word	Vector
<i>jalan</i>	-2.50172567e+00, -1.02271485e+00, 3.24512750e-01, 8.21203768e-01, 2.77345330e-01, -2.61418581e-01, -6.10480928e+00, ..., -1.22443938e+00, -6.80972517e-01, -4.08609390e-01, 1.36039710e+00, -1.08312324e-01, -9.95488167e-01, 1.90991676e+00, ...
<i>padat</i>	-1.4134253, 0.4434754, -1.1915249, -0.5949095, 0.02905422, 0.16940174, -1.3152181, 0.9656274, -0.10436693, 0.71517915, ..., -0.59788555, -0.48828775, -1.0627398, 0.7557978, 0.19303735, 0.15080197, 1.4271564, -0.8112281, -0.0476714, -0.12192555, ...
<i>cakung</i>	-0.17935084, -2.7250762, 1.4563868, -0.11476076, 0.03896635, -0.42993864, 1.867808, -0.4077298, 0.98534757, -0.93986416, ...
<i>rorotan</i>	0.3597058, 0.32056263, -0.9387656, 0.47813663, -1.7315956, -1.467028, 1.4627337, 1.0339308, 0.01472377, 0.5048158, 1.7824717, ...
<i>lenggang</i>	



#### 4.5. Classification results

The data that has been converted to vector is divided into 60% training data, 20% validation data, and 20% test data. Then the data is classified using the Keras library using the CNN method. Before doing the training, the researcher performed CNN hyperparameters to get the best parameters that were tested with training data. The CNN hyperparameters are carried out using Grid Search Cross-Validation, which is to tests the combinations one by one and validate for each combination. The parameters tested can be seen in Table 7.

Table 7. CNN Hyperparameter parameters used

Parameter	Tested Value
<i>learn_rate</i>	0.1, 0.2, and 0.3
<i>dropout_rate</i>	0.1, 0.2, and 0.3
<i>neurons</i>	16, 32, 64, and 128
<i>activation</i>	"relu"
<i>strides</i>	1, 2, and 3

Table 8. Model hyperparameter results

No	Parameter				Model	
	Learning Rate	Neurons	Dropout Rate	Strides	CNN+Word2Vec	CNN+FastText
1	0.1	16	0.1	1	0.8607	0.8673
2	0.2	32	0.1	2	0.8601	0.8697
3	0.3	128	0.1	1	0.8739	0.8373
4	0.1	16	0.2	2	0.8601	0.8643
5	0.2	64	0.2	1	0.8109	<b>0.8775</b>
6	0.3	64	0.2	1	<b>0.8817</b>	0.8493
7	0.3	128	0.2	1	0.8787	0.8439
8	0.1	16	0.3	2	0.8535	0.8577
9	0.3	128	0.3	2	0.8577	0.7832
10	0.3	128	0.3	3	0.8631	0.8571

Based on Table 8 hyperparameters on the CNN+Word2Vec model, the accuracy with training data is 0.8817 or 88.17% with a learning rate of 0.3 neurons 64, dropout rate 0.2 and strides 1 while CNN+FastText is 0.8775 or 87.75% with a learning rate parameter of 0.2, neurons 64, dropout rate 0.2 and strides 1. By using the parameters that have been obtained from the hyperparameter, then training is carried out for 50 epochs using training data, and then for each epoch is evaluated using validation data, the results of training and validation with the CNN+Word2Vec method are 89.08% and 87.38% while in the CNN+FastText method are 87.88% and 87.38%. The validation results are similar from the two models because the data is less and less varied.

#### 4.6. Evaluation results

In this study, the evaluation phase is carried out by comparing the results of the confusion matrix between the CNN+Word2Vec, CNN+FastText, and SVM methods using test data. The SVM model used is taken from the Sklearn library, then the results are entered into a confusion matrix to get accuracy, precision, recall, and f1-score. Table 9 is confusion matrix evaluation using test data by CNN+Word2Vec, CNN+FastText, SVM.

Based on Table 10 or Figure 3, the highest accuracy results were obtained by the CNN+FastText method of 0.8633 or 86.33%, while the lowest value was obtained by SVM of 0.6762 or 67.62%. Then, it was evaluated by comparing the predicted results with the actual congestion manually. The test was carried out 30 times consisting of 15 times the data was stuck and 15 times the data was smooth from a different place or time. Table 11 is the result of the actual evaluation manually.

Table 9. Confusion matrix evaluation using test data

		Actual Value					
		CNN+Word2Vec		CNN+FastText		SVM	
Predicted Value	Congestion	232	59	234	57	180	111
	Smooth	20	245	19	246	69	196

Table 10. Calculation based on confusion matrix with data testing

	CNN+Word2Vec	CNN+FastText	SVM
Accuracy	0.8579	0.8633	0.6762
Precision	0.8059	0.8118	0.6384
Recall	0.9245	0.9283	0.7396
F1-score	0.8611	0.9661	0.6853

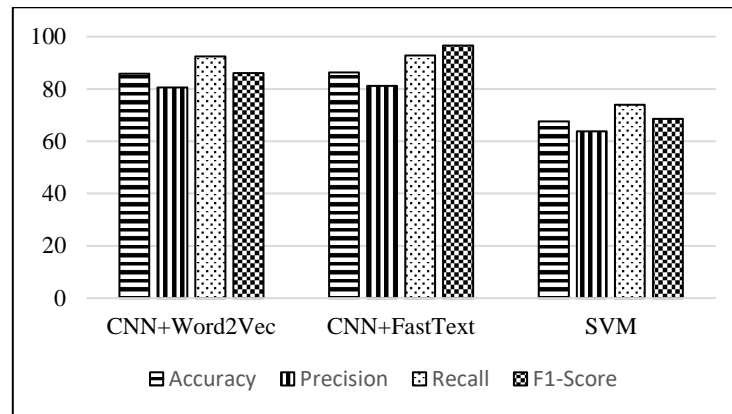


Figure 3. Performance results with test data

Table 11. Result confusion matrix manual actual evaluation

	TP	TN	FP	FN	Accuracy	Precision	Recall	F1-Score
CNN+Word2Vec	11	10	5	4	0.7000	0.6875	0.7333	0.7097
CNN+FastText	11	10	5	4	0.7000	0.6875	0.7333	0.7097
SVM	8	11	4	7	0.6333	0.6667	0.5333	0.5926

Based on Table 11 or Figure 4 the highest accuracy value was obtained by CNN+Word2Vec and CNN+FastText methods which reached 0.7 or 70%, while the lowest accuracy was obtained by SVM of 0.6333 or 63.33%. Based on this research, traffic detection using the CNN method can be applied as supporting data or congestion detection on roads that are not supported by navigation applications, view traffic jam history by date, and can also be implemented in applications for devices that do not have GPS.

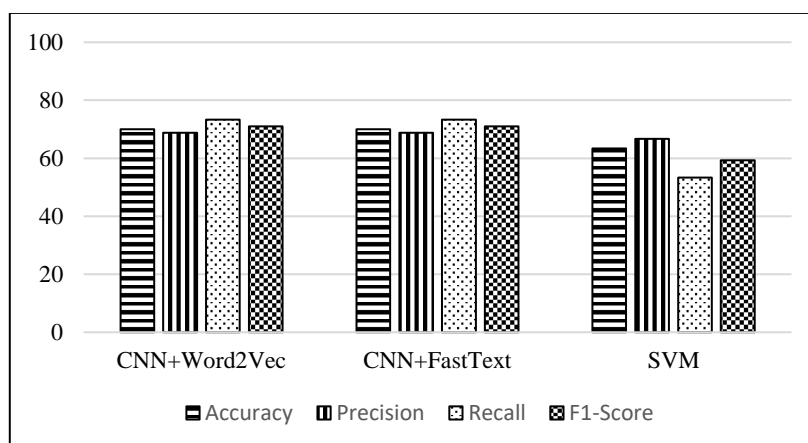


Figure 4. Evaluation results with manual testing

## 5. CONCLUSION

Based on the results of this study, it can be concluded that making a model to detect congestion using tweets takes several stages such as data collection, preprocessing, and data classification. Before

training can be added hyperparameters to get the best parameters for the model. Then, in this research, the CNN+FastText model at the time of evaluation using data testing has the highest level of accuracy compared to CNN+Word2Vec and the SVM method gets the lowest accuracy, while the actual manual evaluation of CNN+FastText and CNN+Word2Vec gets the highest accuracy. For further research, it is recommended to perform a combination of classification methods to increase the accuracy of congestion detection using tweet data. This study only focuses on the classification of tweet data so that further researchers can implement it in the form of an application. Users can directly use the congestion detection application. Then, the dataset can be multiplied to get variations of tweets so that the results are more optimal and in collecting data it is better to involve more than one person to avoid human errors.




## REFERENCES

- [1] L. C. Yang, B. Selvaretnam, P. K. Hoong, I. K. T. Tan, E. K. Howg, and L. H. Kar, "Exploration of road traffic Tweets for congestion monitoring," *Journal of Telecommunication, Electronic and Computer Engineering*, vol. 8, no. 2, pp. 141–145, 2016.
- [2] Statista, "Twitter: most users by country," *Statista*. 2021, [Online]. Available: <https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/>.
- [3] U. Masud, F. Jeribi, M. Alhameed, A. Tahir, Q. Javaid, and F. Akram, "Traffic congestion avoidance system using foreground estimation and cascade classifier," *IEEE Access*, vol. 8, pp. 178859–178869, 2020, doi: 10.1109/ACCESS.2020.3027715.
- [4] Subhan Subhan, E. Sediyo, and Farikhin Farikhin, "The semantic analysis of Twitter data with generative lexicon for the information of traffic congestion," *Journal of Advances in Information Systems and Technology*, vol. 1, no. 1, pp. 45–54, 2019.
- [5] S. Lal, L. Tiwari, R. Ranjan, A. Verma, N. Sardana, and R. Mourya, "Analysis and classification of crime Tweets," *Procedia Computer Science*, vol. 167, pp. 1911–1919, 2020, doi: 10.1016/j.procs.2020.03.211.
- [6] G. A. Ruz, P. A. Henriquez, and A. Mascareño, "Sentiment analysis of Twitter data during critical events through Bayesian networks classifiers," *Future Generation Computer Systems*, vol. 106, pp. 92–104, May 2020, doi: 10.1016/j.future.2020.01.005.
- [7] Ankit and N. Saleena, "An ensemble classification system for Twitter sentiment analysis," *Procedia Computer Science*, vol. 132, pp. 937–946, 2018, doi: 10.1016/j.procs.2018.05.109.
- [8] M. Z. Ansari, M. B. Aziz, M. O. Siddiqui, H. Mehra, and K. P. Singh, "Analysis of political sentiment orientations on Twitter," *Procedia Computer Science*, vol. 167, pp. 1821–1828, 2020, doi: 10.1016/j.procs.2020.03.201.
- [9] M. T. Zulfikar and Suhajito, "Detection traffic congestion based on twitter data using machine learning," *Procedia Computer Science*, vol. 157, pp. 118–124, 2019, doi: 10.1016/j.procs.2019.08.148.
- [10] S. Alhumoud, "Twitter analysis for intelligent transportation," *Computer Journal*, vol. 62, no. 11, pp. 1547–1556, Nov. 2019, doi: 10.1093/comjnl/bxy129.
- [11] M. F. Ahmed, L. Vanajakshi, and R. Suriyanarayanan, "Real-time traffic congestion information from tweets using supervised and unsupervised machine learning techniques," *Transportation in Developing Economies*, vol. 5, no. 2, p. 20, Oct. 2019, doi: 10.1007/s40890-019-0088-2.
- [12] S. Dabiri and K. Heaslip, "Developing a Twitter-based traffic event detection model using deep learning architectures," *Expert Systems with Applications*, vol. 118, pp. 425–439, Mar. 2019, doi: 10.1016/j.eswa.2018.10.017.
- [13] A. K. Sharma, S. Chaurasia, and D. K. Srivastava, "Sentimental short sentences classification by using CNN deep learning model with fine tuned Word2Vec," *Procedia Computer Science*, vol. 167, pp. 1139–1147, 2020, doi: 10.1016/j.procs.2020.03.416.
- [14] E. D'Andrea, P. Ducange, B. Lazzerini, and F. Marcelloni, "Real-time detection of traffic from twitter stream analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 4, pp. 2269–2283, Aug. 2015, doi: 10.1109/TITS.2015.2404431.
- [15] Y. Gu, Z. Qian, and F. Chen, "From Twitter to detector: real-time traffic incident detection using social media data," *Transportation Research Part C: Emerging Technologies*, vol. 67, pp. 321–342, Jun. 2016, doi: 10.1016/j.trc.2016.02.011.
- [16] S. Wongcharoen and T. Senivongse, "Twitter analysis of road traffic congestion severity estimation," in *2016 13th International Joint Conference on Computer Science and Software Engineering, JCSSE 2016*, Jul. 2016, pp. 1–6, doi: 10.1109/JCSSE.2016.7748850.
- [17] Y. Gong, F. Deng, and R. O. Sinnott, "Identification of (near) real-time traffic congestion in the cities of Australia through twitter," in *UCUI 2015 - Proceedings of the ACM 1st International Workshop on Understanding the City with Urban Informatics, co-located with CIKM 2015*, Oct. 2015, pp. 7–12, doi: 10.1145/2811271.2811276.
- [18] G. R. Septianto, F. F. Mukti, M. Nasrun, and A. A. Gozali, "Jakarta congestion mapping and classification from Twitter data extraction using tokenization and naïve bayes classifier," in *Proceedings - APMediaCast: 2015 Asia Pacific Conference on Multimedia and Broadcasting*, Apr. 2015, pp. 14–19, doi: 10.1109/APMediaCast.2015.7210266.
- [19] F. H. Khan, S. Bashir, and U. Qamar, "TOM: Twitter opinion mining framework using hybrid classification scheme," *Decision Support Systems*, vol. 57, no. 1, pp. 245–257, Jan. 2014, doi: 10.1016/j.dss.2013.09.004.
- [20] W. Yao and S. Qian, "From Twitter to traffic predictor: next-day morning traffic prediction using social media data," *Transportation Research Part C: Emerging Technologies*, vol. 124, p. 102938, Mar. 2021, doi: 10.1016/j.trc.2020.102938.
- [21] E. Alomari, R. Mehmood, and I. Katib, "Sentiment analysis of arabic tweets for road traffic congestion and event detection," in *EAI/Springer Innovations in Communication and Computing*, 2020, pp. 37–54.
- [22] Z. Zhang, Q. He, J. Gao, and M. Ni, "A deep learning approach for detecting traffic accidents from social media data," *Transportation Research Part C: Emerging Technologies*, vol. 86, pp. 580–596, Jan. 2018, doi: 10.1016/j.trc.2017.11.027.
- [23] A. Essien, I. Petrounias, P. Sampaio, and S. Sampaio, "A deep-learning model for urban traffic flow prediction with traffic events mined from Twitter," *World Wide Web*, vol. 24, no. 4, pp. 1345–1368, Jul. 2021, doi: 10.1007/s11280-020-00800-3.
- [24] F. Ali, A. Ali, M. Imran, R. A. Naqvi, M. H. Siddiqi, and K. S. Kwak, "Traffic accident detection and condition analysis based on social networking data," *Accident Analysis and Prevention*, vol. 151, p. 105973, Mar. 2021, doi: 10.1016/j.aap.2021.105973.
- [25] E. Alomari, I. Katib, and R. Mehmood, "Iktishaf: a big data road-traffic event detection tool using Twitter and spark machine learning," *Mobile Networks and Applications*, 2020, doi: 10.1007/s11036-020-01635-y.
- [26] D. R. Utari, M. Wibowo, and A. A. Sobari, "Natural language processing of Twitter data for presenting road and traffic information (in Bahasa)," in *Senamika*, 2021, no. April, pp. 756–765.
- [27] J. Salazar-carrillo, M. Torres-ruiz, C. A. Davis, R. Quintero, M. Moreno-ibarra, and G. Guzmán, "Traffic congestion analysis based on a web-gis and data mining of traffic events from twitter," *Sensors*, vol. 21, no. 9, p. 2964, Apr. 2021, doi: 10.3390/s21092964.




- [28] E. Utami, A. D. Hartanto, S. Adi, I. Oyong, and S. Raharjo, "Profiling analysis of DISC personality traits based on Twitter posts in Bahasa Indonesia," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 2, pp. 264–269, Feb. 2022, doi: 10.1016/j.jksuci.2019.10.008.
- [29] Y. Kim, "Convolutional neural networks for sentence classification," 2014, doi: 10.3115/v1/d14-1181.
- [30] P. F. Muhammad, R. Kusumaningrum, and A. Wibowo, "Sentiment analysis using Word2vec and long short-term memory (LSTM) for Indonesian hotel reviews," *Procedia Computer Science*, vol. 179, pp. 728–735, 2021, doi: 10.1016/j.procs.2021.01.061.

## BIOGRAPHIES OF AUTHORS



**Rifqi Ramadhani Almassar**    is currently an employee at Maybank Indonesia, Jakarta since 2022. He got bachelor's in informatics from Universitas Internasional Semen Indonesia, Indonesia, in 2020. He possessed his Master of Informatics (MTI) from University Bina Nusantara, Indonesia, in 2022. He was a programmer at Sinergi Informasi Semen Indonesia from 2019 to 2020, and then joined programmer at Global Service Indonesia from 2020 to 2021. His area of work interest is in the field of Machine Learning, Web Programming, Mobile Programming, Internet of Things, and Artificial Intelligence. He can be contacted at email: rifqi.almassar@binus.ac.id.



**Abba Suganda Girsang**    is currently a lecturer at Master in Computer Science, Bina Nusantara University, Jakarta, Indonesia Since 2015. He got Ph.D. degree in 2015 at the Institute of Computer and Communication Engineering, Department of Electrical Engineering, National Cheng Kung University, Tainan, Taiwan. He graduated bachelor from the Department of Electrical Engineering, Gadjah Mada University (UGM), Yogyakarta, Indonesia, in 2000. He then continued his master's degree in the Department of Computer Science at the same university in 2006–2008. He was a staff consultant programmer in Bethesda Hospital, Yogyakarta, in 2001 and also worked as a web developer in 2002–2003. He then joined the faculty of the Department of Informatics Engineering in Janabadra University as a lecturer in 2003–2015. His research interests include Swarm, Intelligence, Combinatorial Optimization, and Decision Support System. Email: agirsang@binus.edu