DETECTING SUICIDAL IDEATION OF TWEETS USING LONG SHORT-TERM MEMORY AND FASTTEXT WORD EMBEDDING

ALVIN CHRISTIAN NATAPUTRA

DEPARTMENT OF STATISTICS
FACULTY OF MATHEMATICS AND NATURAL SCIENCES
IPB UNIVERSITY
BOGOR
2023

PERNYATAAN MENGENAI SKRIPSI DAN SUMBER INFORMASI SERTA PELIMPAHAN HAK CIPTA

Dengan ini saya menyatakan bahwa skripsi dengan judul *Detecting Suicidal Ideation of Tweets Using Long Short-Term Memory and Fasttext Word Embedding* adalah karya saya dengan arahan dari dosen pembimbing dan belum diajukan dalam bentuk apa pun kepada perguruan tinggi mana pun. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan maupun tidak diterbitkan dari penulis lain telah disebutkan dalam teks dan dicantumkan dalam Daftar Pustaka di bagian akhir skripsi ini.

Dengan ini saya melimpahkan hak cipta dari karya tulis saya kepada Institut Pertanian Bogor.

Bogor, Agustus 2023

Alvin Christian Nataputra
G14190047

ABSTRAK

ALVIN CHRISTIAN NATAPUTRA. Deteksi *Tweets* dengan Pemikiran Bunuh Diri Menggunakan *Long Short-Term Memory* dan *Fasttext Word Embedding*. Dibimbing oleh I MADE SUMERTAJAYA dan GERRY ALFA DITO.

Bunuh diri merupakan krisis kesehatan global. Bunuh diri telah menjadi penyebab kematian terbesar kedua bagi kelompok umur 15 hingga 29 tahun. Di Indonesia, ada sekitar lima korban bunuh diri setiap hari. Untuk mencegah tragedi bunuh diri dalam berbagai kelompok berisiko tinggi, deteksi dini terhadap perilaku bunuh diri dan intervensi yang tepat dan cekatan menjadi sangat penting. Generasi muda beralih ke Internet untuk mencari bantuan dan mendiskusikan topik terkait depresi dan bunuh diri. Besarnya jumlah data teks yang dihasilkan oleh pengguna di jejaring sosial menjadi komponen utama dalam membangun alat deteksi dini. LSTM memiliki performa lebih baik dibandingkan dengan CNN dalam melakukan klasifikasi teks. Fasttext dapat menangani kata-kata yang tidak umum digunakan, kata-kata yang salah eja, serta kata berimbuhan dengan baik. Penelitian ini bertujuan untuk membangun model klasifikasi teks menggunakan LSTM dan fasttext untuk mengidentifikasi *tweet* yang mengandung pemikiran bunuh diri. Tanpa menerapkan metode pra-pemrosesan teks dan penanganan kelas yang tidak seimbang, model telah memiliki performa yang baik dengan sensitivitas 78%, spesifisitas 97%, dan skor F1 88%. Seluruh teknik pra-pemrosesan teks yang digunakan dalam penelitian ini tidak dapat meningkatkan skor F1. Namun, peningkatan sensitivitas tercapai melalui implementasi pembobotan kelas dan *oversampling*. Terutama, teknik ADASYN menghasilkan peningkatan sensitivitas yang signifikan.

Kata kunci: fasttext, klasifikasi teks, LSTM, pemikiran bunuh diri, twitter.

ABSTRACT

ALVIN CHRISTIAN NATAPUTRA. Detecting Suicidal Ideation of Tweets Using Long Short-Term Memory and Fasttext Word Embedding. Supervised by I MADE SUMERTAJAYA and GERRY ALFA DITO.

Suicide is a global health crisis. It has become the second greatest cause of mortality for people aged 15 to 29. In Indonesia, around five people commit suicide every day. To avert the tragedy of suicide in diverse high-risk groups, early detection of suicidal behaviors and adequate and timely interventions are essential. The younger generation has started to turn to the Internet to seek help and discuss depression and suicide-related topics. The huge amount of textual data generated by users on SNS became the main component in building the early detection tool. For a binary text classification task, LSTM Performs better compared to CNN. Fasttext word embedding can handle uncommon words, misspelled words, and word suffixes and prefixes. This research aims to build an accurate text classification model using LSTM and fasttext to identify tweets containing suicidal ideation. Without applying any text preprocessing or imbalanced class treatments, the model had outstanding performance with a 78% sensitivity, 97% specificity, and an 88% F1 score. No text preprocessing technique led to an improvement in the F1 score. However, improvements in sensitivity were achieved through the implementation of class weighting and oversampling. Notably, the ADASYN technique yielded a substantial sensitivity increase.

Keywords: fasttext, LSTM, suicidal ideation, text classification, twitter.

DETECTING SUICIDAL IDEATION OF TWEETS USING LONG SHORT-TERM MEMORY AND FASTTEXT WORD EMBEDDING

ALVIN CHRISTIAN NATAPUTRA

Undergraduate Thesis
to complete the requirement for graduation of
Bachelor Degree in
Statistics and Data Science Study Program

DEPARTMENT OF STATISTICS
FACULTY OF MATHEMATICS AND NATURAL SCIENCES
IPB UNIVERSITY
BOGOR
2023

Tim Penguji pada Ujian Skripsi:
Rahma Anisa, S.Stat., M.Si.

Title : Detecting Suicidal Ideation of Tweets Using Long Short-Term Memory and Fasttext Word Embedding
Name : Alvin Christian Nataputra
NIM : G14190047

Approved by

Main Supervisor:
Dr. Ir. I Made Sumertajaya, M.Si.

Co-Supervisor:
Gerry Alfa Dito, S.Si., M.Si.

Acknowledged by

Head of Statistics Department:
Dr. Bagus Sartono, S.Si., M.Si.
NIP. 19780411 200501 1002

Thesis Defense Date:
August 4, 2023

Graduation Date:

PREFACE

The author extends sincere gratitude to the Almighty God for His unconditional love and unending blessing so that this research can be completed. The topic of the research that has been conducted since February 2023 to July 2023 is building text classification model to detect the presence of suicidal ideation within tweets, titled "Detecting Suicidal Ideation of Tweets Using Long Short-Term Memory and Fasttext Word Embedding".

The completion of this scientific paper is due to various support and assistance from different parties. Therefore, the author expresses appreciation to:

1. Dr. Ir. I Made Sumertajaya, M.Si. and Gerry Alfa Dito, S.Si., M.Si. for their invaluable guidance and insights throughout the research journey;
2. The Lecturers and Administrative Staff in the Department of Statistics for fostering an environment conducive to academic growth;
3. Author's family and friends for their unwavering support and encouragement;
4. All individuals and entities that have given all kinds of assistance in various way so that this research can be completed.

Hopefully, this research can be useful to many people and for a variety kind of purposes. Acknowledging the potential limitations and imperfections of this research, the author welcomes constructive feedback and recommendations with open arms. Such input will be greatly appreciated and duly considered.

Bogor, August 2023

Alvin Christian Nataputra

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

I INTRODUCTION

1.1 Background

Suicide is a major global health issue. According to the WHO (2021), there are approximately 700.000 suicide victims worldwide, making it the fourth-leading cause of death among those aged 15 and 29. The National Institute of Health Research and Development in Indonesia revealed that there were 1.800 suicide cases reported in 2016, which corresponds to about five suicide victims per day (KEMENKES 2021). However, suicide is preventable. Early identification of suicidal tendencies and appropriate and timely interventions are crucial to preventing the tragedy of suicide in a variety of high-risk groups (Que *et al*. 2020). In a meta-analysis conducted by Franklin *et al.* (2017), the number-one risk factor for future episodes of suicidal ideation was prior suicidal ideation. Globally, lifetime prevalence rates are approximately 9,2% for suicidal ideation and 2,7% for suicide attempt (Nock *et al.* 2008).

Social Networking Sites (SNS) are virtual platforms that enable users to interact with one another through user-generated content (Wang *et al.* 2015). To this date, SNS is one of the most popular and fastest-growing communication technologies worldwide (Tsai *et al.* 2017). One of the most popular SNS is none other than Twitter. As of the second quarter of 2022, Twitter had on average 237,8 million daily active users. Twitter finds popularity amongst those aged 13 to 34, with this age group accounting for up to 60% of the social platform's worldwide user base in 2022 (Dixon 2023). The younger generation has begun to use the Internet to discuss themes linked to depression and suicide as well as to seek treatment (Chan dan Fang 2007). The huge amount of textual data generated by the users on SNS became the main component in building the early detection tool. The major practical application of this tool lies in its easy adaptability to any social media forum (Robinson *et al.* 2016).

Text mining is defined as the process of extracting the implicit knowledge from textual data (Feldman dan Sanger 2008). One of text mining technique is text classification. The vast amount of textual data generated makes it almost impossible for human to perform manual text classification which also would take huge amount of time and resources in the process. Automatic text classification is the process of automatically assigning a text or a document to a set of pre-defined classes using a machine learning technique (Dalal and Zaveri 2011). The development of technology has seen an increase in the use of machine learning and deep learning algorithms to solve text classification problem. Long Short-Term Memory (LSTM) is capable of capturing long-term dependencies in data and has been shown to be effective in detecting patterns in text. LSTM is a modified version of the Recurrent Neural Network (RNN). LSTM is able to solve the exploding or vanishing gradient problem that exists in RNN. Faadilah (2020) performed sentiment analysis of Tokopedia's review on Google Playstore with LSTM resulting in 93.32% accuracy. Previous research has compared the performance of Convolutional Neural networks (CNN) and LSTM in text classification and shown that LSTM performs better compared to CNN (Adina 2020). Automatic text classification becomes the main engine to build the early identification tool that works by crawling through tweets and flag the tweets that are detected as having presence of suicidal ideation.

In terms of word embedding technique, research by (Tiun *et al.* 2020) concluded that fasttext results were superior compared to word2vec, especially for a binary classification with a very short length of text and the least number of vocabularies. Riza and Charibaldi (2021) compared the performance of glove and Fasttext word embedding using LSTM model shown that fasttext has the advantage of handling the out of vocabulary (OOV) problem and thus produce better accuracy (73.15%) compared to the glove (60.10%).

Therefore, developing an early identification tool to detect tweets containing suicidal ideation and providing them with the necessary interventions could help possible suicide victims. This research could contribute to supporting Sustainable Development Goal (SDG) number 3. SDG number 3 aims to ensure healthy lives and promote well-being for all at all ages. One of its targets is to reduce premature mortality from non-communicable diseases through prevention and treatment and promote mental health and well-being which includes cardiovascular disease, cancer, diabetes, chronic respiratory disease, and suicide (UNSC 2016).

1.2 Objectives

This research aims to develop a binary text classification model by utilizing the fasttext word embedding method and LSTM neural network architecture to identify tweets containing suicidal ideation. Moreover, this research investigates the varying performance outcomes of the proposed model under diverse scenarios of imbalanced class treatment and various text preprocessing applications for this specific study case.

II LITERATURE REVIEW

2.1 Suicidal Ideation

Suicidal ideation, often called suicidal thoughts or ideas, is a broad term used to describe a range of contemplations, wishes, and preoccupations with death and suicide (Harmer *et al.* 2020). Klonsky *et al.* (2016) defined suicidal ideation as thinking about, considering, or planning suicide. Suicide ideation and attempts are strongly predictive of suicide deaths. It can also result in negative consequences such as injury and hospitalization (Nock *et al.* 2008). Based on the definition of suicidal ideation, Sawhney *et al.* (2018) constructed a set of rules to define the presence of suicidal ideation in a tweet. The adapted rules are shown in Table 1.

Table 1 Data labeling rules

Presence of suicidal ideation (1)	Absence of suicidal ideation (0)
Tweets convey an explicit or serious display of suicidal ideation	Default category for all tweets that have no presence or relevance to suicidal ideation
Tweets where suicide plan and/or previous attempts are discussed	Tweets emphasizing on suicide related news or information
Tweets asking for advice or ways to hurt or kill himself/herself	Tweets pertaining to condolence and suicide awareness
Tweets implicitly stating the intent to die/end their life if a condition is fulfilled and displaying no will to continue living	Tweets containing song lyrics, movies title/quotes, or discussion about fictional works

2.2 Text Preprocessing

Text preprocessing is a step in text mining in which the textual data that has been gathered are processed through several methods. Vijayarani *et al.* (2015) stated that the preprocessing method plays a very important role in text mining techniques and applications. Preprocessing involves transforming text prior to analysis by identifying which units to use, removing irrelevant content for some tasks (i.e., removing nonalphabetic characters and stop words), agglomerating semantically related terms to reduce data sparsity and improve the model's predictive power (i.e., converting all letters to lowercase, expanding contractions and abbreviations, and stemming or lemmatizing), and increasing the amount of semantic information that is captured (i.e., handling negation). However, this means that preprocessing can also remove useful information (e.g., removing stop words when they are relevant to the research question), introduce errors into analysis (e.g., when stemming conflates semantically distinct words), and drastically alter subsequent results (Hickman *et al.* 2022).

The text preprocessing methods that will be used in this research are case folding, stemming, removing stop words, removing mentions and hashtags,

removing hyperlinks, removing punctuation, removing special characters, and removing redundant whitespace.

2.2.1 Case Folding

Case folding is the first step in performing text preprocessing methods. It turns every single letter in the textual dataset into lower case (Luqyana *et al.* 2018). Case folding can solve the double indexing problem where the exact same words are indexed differently because of different cases.

2.2.2 Mention and Hashtag Removal

Tweets can contain mentions and hashtags. Mention is a part of the tweet in which the user wants to mention or reply to another Twitter user. It is identified as a mention if the word starts with the symbol '@'. For example, in "I love using Twitter thanks to @twitter_id", @twitter_id is considered a mention. A hashtag is a unique tagging convention to help associate tweets with certain events or contexts (Chang 2010). It is prefixed by a '#' symbol. For example, in "I support animal protection #wildlife, #animal_lovers,", the words #wildlife and #animal_lovers are considered hashtags.

2.2.3 Hyperlink Removal

Hyperlink is defined as a technological capability that enables one specific website (or webpage) to link with another (Park 2003). An example of a hyperlink in a tweet is "https://apricitea.github.io" or "https://twitter.com/home". The common components of a hyperlink start with "https://" or "http://", contain domains like ".com", ".io", or ".co.id", and contain information about the path or subdomain separated with the symbol "/". Hyperlinks contain little to no information related to the tweets, and thus performing hyperlink removal could reduce training time without decreasing the model's performance.

2.2.4 Punctuations Removal

Punctuation is a set of standardized marks or symbols used in writing to clarify meaning, indicate the structure of a sentence, and aid in reading comprehension. Examples of punctuation that is going to be removed are the period (.), comma (.), exclamation mark (!), and question mark (?). While the use of punctuation is crucial in retaining the meaning of a sentence, removing punctuation could reduce variability in the textual data. Removing punctuation can also be useful to reduce scenarios when it is incorrectly used. As a result, performing punctuation removal could reduce training time and variability in the dataset.

2.2.5 Special Characters Removal

Special characters are any characters in a text besides alphabetical and numerical characters. In tweets, special characters occur when the user uses emojis or symbols. Removing special characters could reduce the noise and clutter in the data. Examples of special characters that are going to be removed are operator signs (+, -, =, ^), symbols (&, $, @, *), and others (ð, Ÿ, ¥, ™, ©).

2.2.6 Stop Words Removal

Stop words are words that have no important meaning in a sentence and are considered not to have much impact on the classification model's performance (Patel and Shah 2013). Some examples of stop words in Indonesian are "*di*", "*yang*", "*sebuah*", and "*ada*". Removing these stop words could reduce the number of words required to be processed and result in less training time. The Python library that will be used to perform stop word removal in this research is 'sastrawi', which stores a list of stop words in Indonesian.

2.2.7 Stemming

Stemming is the process of reducing words to their base or root form. Two ways to perform stemming are by using a dictionary of words and their root forms and using suffixes and affixes rules (Utomo 2013). Stemming reduces the variance of words in the dataset, which could reduce the embedding matrix size and result in less training time. The python library that will be used to perform stemming in this research is 'hunspell' stemmer for Indonesian.

2.2.8 Redundant Whitespaces Removal

Removing redundant whitespaces is the last step in performing text preprocessing to clean the data. Redundant whitespaces might exist in the initial dataset due to mistyping or user behavior. It can also be caused by performing several text preprocessing methods that remove components of the tweets.

2.3 Long Short-Term Memory (LSTM)

LSTM is a deep learning algorithm developed from the RNN architecture. With the use of memory cells and gate units (input gate, forget gate, and output gate), it can read, store, and update information so that LSTM can address the issue of vanishing/exploding gradients (Rao dan Spasojevic 2016). The forget gate determines which information is to be deleted from the cell. The input gate objective is to determine the input value that will be updated in the state memory. Based on the input and memory in the cell, the output gate determines the output. The following are the LSTM steps (Hochreiter dan Schmidhuber 1997):

- The first step in LSTM, known as the forget gate, will determine which information should be removed from the cell state. The input (x_t) and output from the previous step (h_{t-1}) are used in this step and processed using the sigmoid activation function. This process will output a range of value from 0 to 1, a value of 0 indicates that information will be entirely removed, while a value of 1 indicates that it will be preserved. The forget gate equation is described in equation (1).

$$ft = \sigma(\boldsymbol{x_t} \cdot \boldsymbol{W_f} + \boldsymbol{h_{t-1}} \cdot \boldsymbol{U_f} + \boldsymbol{b_f}) \quad (1)$$

- The information that will be added to the cell state is determined in the second step. This step consists of two parts; the first is the input gate (i_t), which functions to determine the importance of the new candidate value to be remembered. The second part is the tanh layer that generate a new candidate value ($\check{C}_t$) to be added into the cell state. The output from the input gate layer and tanh layer will be combined to update the cell state.

The equation for the input gate layer and the equation for the candidate value can be seen in equation (2) and equation (3)

$$i_t = \sigma(\boldsymbol{x_t} \cdot \boldsymbol{W_i} + \boldsymbol{h_{t-1}} \cdot \boldsymbol{U_i} + \boldsymbol{b_i}) \quad (2)$$

$$\check{C}_t = \tanh(\boldsymbol{x_t} \cdot \boldsymbol{W_c} + \boldsymbol{h_{t-1}} \cdot \boldsymbol{U_c} + \boldsymbol{b_c}) \quad (3)$$

- In the third step, the old state cell (C_{t-1}) is updated to become the new state cell (C_t). The new cell state (C_t) is obtained by summing the multiplication of input gate value (i_t) and the candidate context ($\check{C}_t$) that were obtained in the previous step with the multiplication of forget gate value (f_t) and the old state cell (C_{t-1}). The equation for getting a new cell state (Ct) can be seen in equation (4).

$$C_t = f_t \times C_{t-1} + i_t \times \check{C}_t \quad (4)$$

- The outcome of the entire process is determined by the output value (h_t) obtained in the last step. Similar to the forget gate equation, the input (x_t) and output from the previous step (h_{t-1}) are processed using the sigmoid activation function to obtain the gate output value (O_t). Next, the cell state (C_t) is processed through the tanh activation function. It will output a value between -1 and 1. The cell state value ($tanh\ (C_t)$) is multiplied by the gate output value (O_t) to get the output value (h_t). This process equation is as described in equation (5) and equation (6).

$$O_t = \sigma(\boldsymbol{x_t} \times \boldsymbol{W_o} + \boldsymbol{h_{t-1}} \times \boldsymbol{U_o} + \boldsymbol{b_o}) \quad (5)$$

$$h_t = O_t \times tanh\ (C_t) \quad (6)$$

- Dense layer is used to determine the classification results. The dense layer gets input from the final output (h_t) of the last order. The Dense Layer formula can be seen in equation (7).

$$Y = \sigma(\boldsymbol{h_t} \times \boldsymbol{W_y} + \boldsymbol{b_y}) \quad (7)$$

where $\sigma(x)$ represent sigmoid activation function, $\tanh(x)$ represents hyperbolic tangent activation function, $\boldsymbol{x_t}$ represents input at the current timestamp, $\boldsymbol{h_{t-1}}$ represents output from previous LSTM block, $\boldsymbol{W}_x$ and $\boldsymbol{U_x}$ represent the weight for the respective gate(x) neurons, and $\boldsymbol{b_x}$ represent bias for each respective gate(x).

To build an LSTM model, hyperparameters are used. Hyperparameter is a deep learning element that is not updated in the training phase but affects the model's performance (Han *et al.* 2020). There are several hyperparameters that will be used in this model:

2.3.1 Epochs

An epoch is defined as the period in which each training sample is used once for updating the model parameters (Kratzert *et al.* 2018). One epoch means the model has seen each training sample once. Training usually involves multiple epochs to improve the model's performance. Larger epochs could improve the model's convergence, but too many epochs may lead to overfitting.

2.3.2 Batch Size

During training, data is divided into smaller subsets called batches (Kratzert *et al.* 2018). Batch size determines the number of samples in each batch. It is a hyperparameter that affects the speed and quality of training. Larger batch sizes can speed up training but require more memory, while smaller batch sizes can provide more frequent updates to the model but might be slower overall.

2.3.3 Dropout

Dropout is a regularization technique used to prevent overfitting in neural networks. The term "dropout" refers to dropping out units (hidden units and input units) in a neural network. The choice of which units to drop is random. By dropping a unit out along with all its incoming and outgoing connections, the model becomes more robust and generalize better to new data. The fraction of units to be dropped is a hyperparameter and is usually set between 0.2 and 0.5 (Srivastava *et al.* 2014).

2.3.4 Recurrent Dropout

Recurrent dropout helps regularize the LSTM and prevent overfitting, specifically in the temporal context. Recurrent dropout is a variation of the dropout technique specifically tailored for RNNs, including LSTM networks. In addition to applying dropout to the input units as in the regular dropout technique, recurrent dropout applies dropout to the recurrent connections within the RNN. In an LSTM network, the recurrent connections play a crucial role in carrying information across time steps. By applying recurrent dropout, a fraction of the recurrent connections (the connections between LSTM units across different time steps) are randomly set to zero during each forward and backward pass of the training process. This helps prevent overfitting and improves the generalization of the LSTM by reducing the co-adaptation of LSTM units across time (Srivastava *et al.* 2014).

2.3.5 LSTM Units

LSTM units (also known as LSTM cells or hidden units) are the individual building blocks of the LSTM layer. They contain the memory cell and the gates responsible for controlling the information flow. The number of LSTM units in a layer determines the capacity of the LSTM to learn complex patterns in the data. Larger LSTM units might lead to better performance but require more training data and resources. The component of each hidden unit is shown in Figure 1.

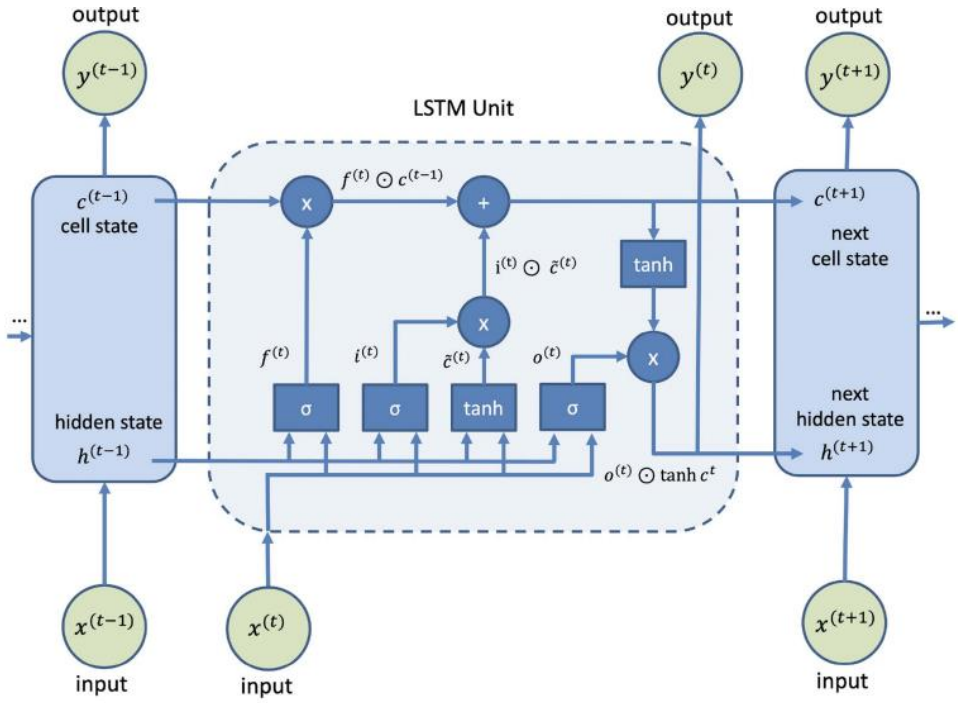


Figure 1 Architecture of an LSTM unit that connects to other units (Lazaris and Prasanna 2021)

2.3.6 Dense Layer

A dense layer, also known as a fully connected layer, is a type of neural network layer where each neuron is connected to every neuron in the previous and following layers. Once the LSTM processes the input sequence and extracts relevant information, it may not be directly usable for the final prediction task. The dense layer acts as a bridge between the LSTM's hidden representation and the final output. It receives the LSTM's output and maps it to the desired output dimension. The purpose of the dense layer is to combine the relevant information learned by the LSTM and convert it into the appropriate output format, such as class probabilities or numerical values.

2.3.7 Activation Function

The activation function introduces non-linearity to the output of a neuron. They allow the neural network to learn complex relationships in the data. The activation functions used in LSTM are the sigmoid activation function and the hyperbolic tangent activation function. The equations for the activation function are as in equation (8) and equation (9):

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (8)$$

$$\tan h(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad (9)$$

where e^{x} represent exponential value of x.

2.3.8 Optimizer

The optimizer is an algorithm used to update the model's weights during training based on the calculated gradients. Popular optimizers include Adam, RMSprop, and SGD (Stochastic Gradient Descent). The 'Adam' optimizer stands for Adaptive Moment Estimation. It combines the advantages of two other optimization techniques: AdaGrad and RMSprop. The key features of the Adam optimizer include adaptive learning rates for each parameter and momentum-like behavior for faster convergence, which enhances the overall model performance. Each time, Adam updates new weights based on the previous value of an attribute (Sharma 2019).

The adaptive learning rates help the model adjust the learning rates for each parameter during training. It allows the optimizer to update the parameters more efficiently, especially for high-dimensional and non-stationary problems like training deep neural networks. The Adam optimizer is known for its robustness, good convergence properties, and ease of use. It often requires less tuning of its hyperparameters compared to other optimization techniques like stochastic gradient descent (SGD).

The algorithm was created by Kingma and Ba (2014). The steps taken to perform the optimization are:

1. The default settings for the tested machine learning problems are stepsize $(\alpha) = 0{,}001$, exponential decay rates for the moment estimates $\beta_1 = 0{,}9$, $\beta_2 = 0{,}999$, and $\epsilon = 10^{-8}$. With g_t represent the gradient of the loss with respect to the parameter at iteration t and g_t^2 indicates the elementwise square $g_t \odot g_t$.
2. Set $m_0 = 0$ as initial 1st moment vector
3. Set $v_0 = 0$ as initial 2nd moment vector
4. Set $t = 0$ as initial timestep
5. Perform looping: while θ_t does not converge do:
   a. $t = t + 1$
   b. $g_t = \nabla_\theta f_t(\theta_{t-1})$
   c. $m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
   d. $v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
   e. $\widehat{m}_t = m_t / (1 - \beta_1^t)$
   f. $\widehat{v}_t = v_t / (1 - \beta_2^t)$
   g. $\theta_t = \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (updating parameter)
6. Return θ_t (the resulting parameter)

where $\nabla_\theta f_t(\theta_{t-1})$ represent the gradient of the loss function with respect to the model's parameters at iteration t. The loss function ($f_t(\theta_{t-1})$) used in this research is the binary cross-entropy loss function.

2.3.9 Loss Function

During neural network training, the cost function is the key to adjusting a neural network's weights to create a better fitting machine learning model (Ho and Wookey 2020). The loss function measures the difference between the

predicted output and the actual target for a given input. It quantifies how well the model is performing on the training data. The goal is to minimize the loss during training. The binary cross-entropy loss function is commonly used in binary classification tasks, where the goal is to classify data into one of two classes. Binary cross-entropy is a special class of cross-entropy in which the target of the prediction is 1 or 0. It is derived from the concept of information entropy and is well-suited for binary classification problems (Ruby *et al.* 2020).

The binary cross-entropy loss quantifies the dissimilarity between the predicted probabilities and the true binary labels. It encourages the model to assign high probabilities to the correct class and low probabilities to the incorrect class.

The standard binary cross-entropy function is given by the equation (10):

$$L = -\frac{1}{M}\sum_{m=1}^{M}[y_m \times \log(h_\theta(x_m)) + (1 - y_m) \times \log(1 - h_\theta(x_m))] \quad (10)$$

where L represents the loss, M represents the number of training examples, y_m the target label for training example m (0 or 1), x_m input for training example m, and h_θ represent model with neural network weights θ (Ho and Wookey 2020).

2.4 Fasttext Word Embedding

Word embedding is a distributional representation of words in which each word is assigned a d-dimensional vector and should be mapped to a common low-dimensional space (Goldberg 2017). The benefit of employing word embeddings over text representation is that the former can capture both the semantic and syntactic links between words, while the latter only captures word frequency. Embedding methods can also reduce the dimensionality of the word space.

Fast text is a word embedding method developed from Word2Vec. Fasttext offers the advantage of handling the out-of-vocabulary (OOV) issue, which cannot be resolved by word2vec and glove word embeddings. Fast text learns word representation by paying attention to sub-word information using n-grams in the skip-gram model. This enables fasttext to capture shorter words, handle uncommon and misspelled words, and comprehend word suffixes and prefixes better. For instance, the words play, played, playing and playful all have similar vector representations, even if they tend to show up in different contexts. Here, each word is represented as character n-gram. So, for n=3, the words play and playful will be represented as: $< pl, pla, lay, ay >$ and $< pl, pla, lay, ayf, yfu, ful, ul >$. This approach preserves subword information and can compute valid word embedding for out-of-vocabulary words (Bojanowski *et al.* 2017).

To preserve the subword information, the developers of fasttext presented alternative scoring systems. Given a word w, the set of n-grams appearing in w will be $N_w \subset \{1, \ldots, N\}$, where N is the dictionary size of n -grams. Vector representation (denoted as Z_g) is assigned for each n-gram, where c = context word, and V_c= context vector. The derived scoring function is shown on equation (11):

$$s(w, c) = \sum_{n \in N_w} \boldsymbol{Z_g^T}\ \boldsymbol{V_c} \quad (11)$$

2.5 Imbalanced Class Treatment

Similar research done by Sawhney *et al.* (2018) collected 5.213 tweets and only 822 (15,76%) of the tweets containing suicidal ideation were found. It suggests that the data collected in this research may have an imbalanced class problem. Class imbalance can occur when the instances of one class outnumber the instances of other classes. In many applications, the class that has fewer instances is the more important one. The class imbalance problem heightens whenever the class of interest is relatively rare and has a small number of instances compared to the majority class. Moreover, the cost of misclassifying the minority class is very high in comparison with the cost of misclassifying the majority class (Elrahman and Abraham 2013).

LSTM learns from the training data by minimizing a loss function, and the weights assigned to each sample influence the magnitude of the loss contribution from that sample. By increasing the weight of the less-represented class, the loss contribution of each sample from that class is effectively amplified. This can lead to better classification performance for the minority class, as the model is forced to learn more discriminative features and decision boundaries to differentiate the less-represented class from the majority class. Class weight appears to reduce the impact of the data imbalance and improve the model's performance (Sun *et al.* 2020). Ahamed *et al.* (2020) achieved the best performance with 98,06% accuracy using class weight during the ANN model training.

Oversampling method is a technique used to address class imbalance by increasing the number of samples in the minority class. Synthetic Minority Over-sampling Technique - Edited Nearest Neighbor (SMOTE-ENN) and Adaptive Synthetic (ADASYN) are some of the few oversampling methods that have better performance compared to the other methods. Sir and Soepranoto (2022) concluded that oversampling methods can solve imbalanced class problem with SMOTEENN and ADASYN ranked 1st and 2nd respectively in their performance. SMOTEENN combines the methods of Synthetic Minority Over-sampling Technique (SMOTE) and Edited Nearest Neighbor (ENN). The synthetic samples are generated by SMOTE to obtain an augmented dataset T, then the number of nearest neighbours as K are determined, after which K examples around the observation sample in the dataset T are selected. The observation sample and the K neighbours are removed if the class of the observation is inconsistent with its K nearest neighbour majority class (Batista *et al.* 2004). The SMOTE and ENN procedures are repeated until the desired number of each class is achieved. ADASYN adaptively synthesizes unequal datasets both for the majority and minority classes to balance the data (He *et al.* 2008). The number of synthetic data examples for the minority class depends on the distribution in ADASYN, which means a weight is assigned adaptively to the minority class. Therefore, the more cases that are labeled with the majority class around the sampling data in the minority class, the more synthetic cases are generated for the sampling data (Wang *et al.* 2023).

2.6 Model Evaluation

To evaluate the model, a confusion matrix and several evaluation metrics will be used to compare the performance of each scenario. Confusion matrix is a tool to analyze a model's performance in recognizing whether the classifier correctly

predicted the class. There are four terms in the confusion matrix that represent the results. They are True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). TP indicated the number of positive data correctly predicted as positive, TN indicated the number of negative data correctly predicted as negative, FP indicated the number of negative data incorrectly predicted as positive, and FN indicated the number of positive data incorrectly predicted as negative. In this specific research, FP is considered a better case compared to FN because the impact of falsely predicting a positive class as negative is higher compared to the reverse. The illustration of the confusion matrix can be seen in Figure 2:

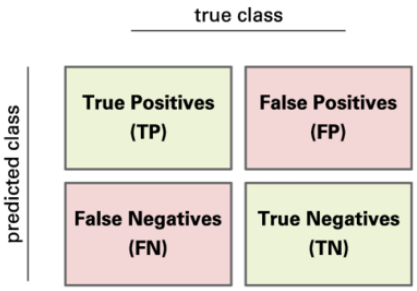


Figure 2 Confusion matrix (Bittrich *et al.* 2019)

Evaluation metrics that will be used to measure the model's performance include: sensitivity, specificity, precision, recall, and f1-score. Sensitivity and specificity are used to explain the model's performance in detecting a specific class. Sensitivity is the ratio of the correctly predicted positive data compared to the actual number of positive classes in the data. Specificity indicates the same value but for the negative classes. The macro-average of recall, precision, and F1 score measured the overall model's performance in detecting both classes. In macro-averaging, the contingency table of each individual class j is used to compute that particular class's precision $precision_j$ as well as recall $recall_j$, and finally compute a simple average of the F1 scores over classes to get macro-average F1. The macro-average of recall, precision, and F1 score will be denoted as $recall_m$, $precision_m$, and $F1\ score_m$. The equations for the metrics are as shown in equation (12), equation (13), equation (14), equation (15), and equation (16) respectively (Dalianis 2018):

- $sensitivity = \frac{TP}{TP+FN}$ (12)
- $specificity = \frac{TN}{TN+FP}$ (13)
- $recall_m = \left(\frac{TN}{TN+FP} + \frac{TP}{TP+FN}\right)/2$ (14)
- $precision_m = \left(\frac{TN}{TN+FN} + \frac{TP}{TP+FP}\right)/2$ (15)
- $F1\ score_m = 2\frac{precision_m \cdot recall_m}{precision_m + recall_m}$ (16)

III METHODOLOGY

3.1 Data

Tweepy is a Python library used in this research to crawl tweets with official Twitter Developer API access. O'Dea *et al.* (2015) constructed a list of words and phrases that are consistent with the vernacular of suicidal ideation. The list is translated into Indonesian and used as keywords in crawling the tweets. The keywords in Indonesian are as follows: *bunuh saya; bunuh diri; ingin bunuh diri; mau bunuh diri; coba bunuh diri; percobaan bunuh diri; cara bunuh diri; pesan bunuh diri; catatan bunuh diri; pikiran bunuh diri; rencana bunuh diri; ide bunuh diri; mengakhiri hidup; akhirin hidup; tidak ingin melanjutkan hidup; tidak ada gunanya hidup; tidak ada alasan untuk hidup; tidak ada artinya hidup; tidak ingin hidup; tidak kuat hidup; melompat bunuh diri; terjun bunuh diri; tidur selamanya; mau mati; ingin mati; lebih baik saya mati; lebih baik mati; mati sendirian; siap untuk mati; ingin mati sekarang; tidak inigin ada di sini; cutting; melukai pergelangan tangan; ingin semuanya selesai; mengambil nyawaku sendiri; depresi; saya berharap saya mati; bunuh saya.* A total of 10.128 tweets crawled from February 8 to May 31, 2023.

The annotators assign the label of the collected tweets as either "presence of suicidal ideation (labeled as 1)" or "absence of suicidal ideation (labeled as 0)". The annotators consisted of three undergraduate students who were fairly active on social media and well-informed about aspects of suicide awareness. To ensure the consistency and reliability of the data labeling process by the annotators, specific rules are made to determine the presence of suicidal ideation in each tweet. The rules were adapted from previous research done by Sawhney *et al.* (2018) as previously shown in Table 1.

To preserve the users' privacy, personal information (e.g., username and display name) is not collected; their personal identification is replaced with a unique index for each of the tweets collected. The final dataset consisted of 10.128 unique tweets and contained three columns, which are the index, the tweets, and their respective labels, is shown in Table 2.

Table 2 Final dataset used in this research

Index	Tweets	Labels
1	help, gue lg pengen mati bgt rasanya, gakuat sm idup, gakuat jd manusia, gakuat punya trauma, mau bunuh diri aja	1
2	nekat mau belajar ukulele agaknya bunuh diri	0
3	orang orang ngeliat aku kaya gaada beban kali ya padahal mah, kalo di tanya sanggup ga jalanin hidup? Jawabannya ngga. Rasa nya putus asa buat jalanin kehidupan ini sampe mau bunuh diri	1
…	…	
10128	"LEBIH 5 TAHUN LEBIH 5 RIBU MAHASISWA BUNUH DIRI #CokroTV #GakPakeBaper #mahasiswa #mentalhealth Full Video Youtube: https://t.co/1aiJYVZxnP https://t.co/TdXynrY3se"	0

3.2 Analysis Procedure

The analysis procedure in this research is shown in Figure 3:

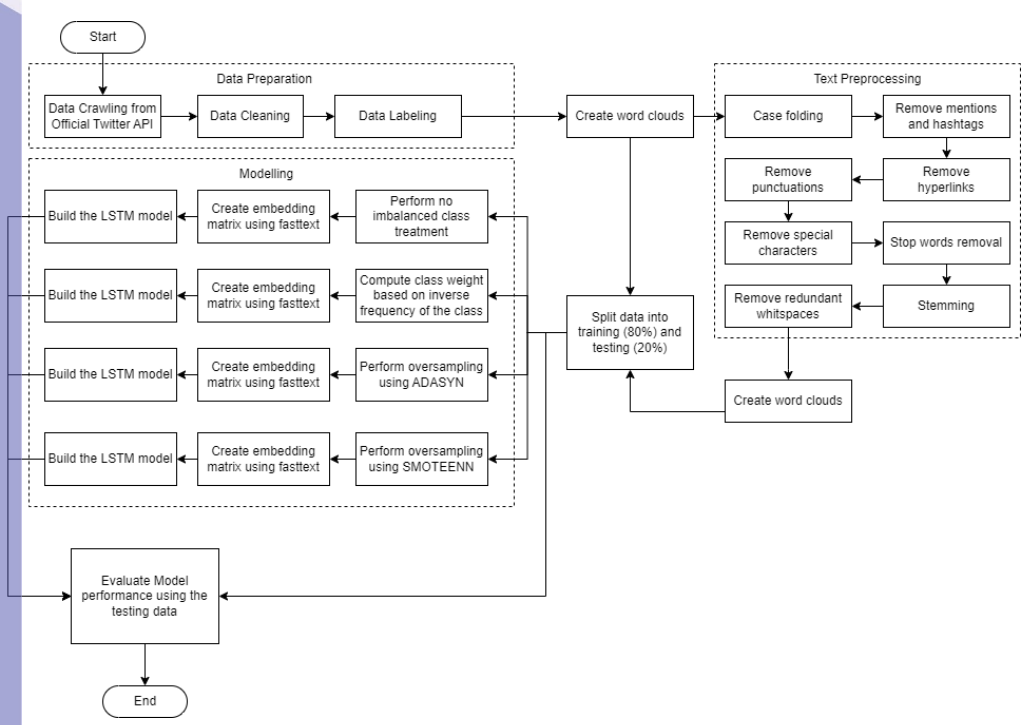


Figure 3 Analysis procedure flowchart

1. Crawl tweets using Tweepy in a Python (version 3.9.13) environment.
2. Perform initial data cleaning, which includes removing duplicate data, removing non-Indonesian tweets, and removing tweets with only hyperlinks or special characters.
3. Three annotators perform data labeling on the obtained tweets according to the rules specified in Table 1, and the final label of each tweet is decided by a majority vote.
4. Perform several text preprocessing methods to the collected tweets, which include: case folding, removing mentions and hashtags, removing hyperlinks, removing punctuations, removing special characters, removing stop words, stemming, and removing redundant whitespace.
5. Create word clouds for tweets prior and posterior to text preprocessing to compare the appearance of words in each class.
6. Split the data into training and testing sets by a ratio of 80:20 respectively.
7. Load the embedding matrix using fasttext word embedding. The resulted word embedding matrix is fed to the LSTM model. This research use the pretrained word vector of Indonesian language provided by fasttext. The model was trained using CBOW with position-weights, in dimension 300, with character n-grams of length 5, a window of size 5 and 10 negatives (Grave *et al.* 2018). The illustration is shown in Figure 4:

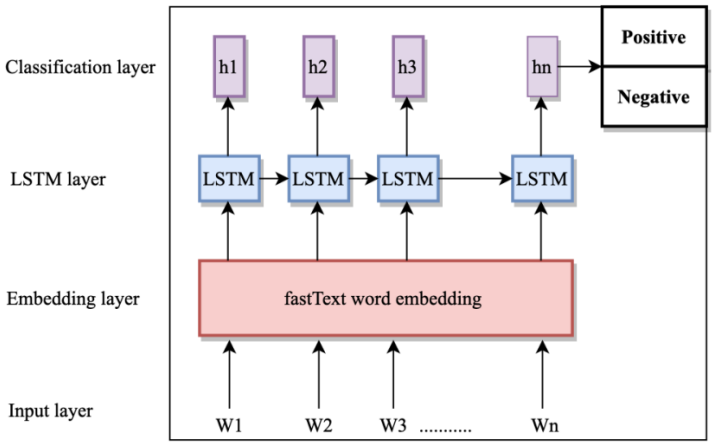


Figure 4 System architecture of an LSTM network with a fasttext word embedding layer based on Khan *et al.* (2021)

8. Build the LSTM model using the training data and the resulted embedding matrix from fasttext with four different scenarios to handle the imbalanced class problem as shown in Figure 3. For the binary classification task to detect the presence of suicidal ideation in tweets, the LSTM model is trained using the training data and the embedding matrix produced by pretrained fasttext word embeddings. Table 3 presents the parameter specification for the proposed LSTM model. The models are built using the 'keras' deep learning framework in Python (version 3.9.13) and the experimental environment is trained on NVIDIA 920MX in a 64-bit computer with Intel(R) Core(TM) i5-8250U CPU @1.6GHz, 8 GB RAM, and Windows 11 operating system.

Table 3 LSTM Model Specification

Parameters	Values
Epochs	10
Batch size	8
Dropout	0,2
Recurrent dropout	0,2
LSTM units	64
LSTM units activation function	'Sigmoid' and 'tanh'
Output layer	1
Output layer activation function	'Sigmoid'
Optimizer	'Adam'
Loss function	'Binary crossentropy'

9. Evaluate the model's performance using confusion matrix and several classification metrics including sensitivity, specificity, precision $_m$, recall $_m$, and f1-score $_m$.

IV RESULT AND DISCUSSION

4.1 Data Preparation

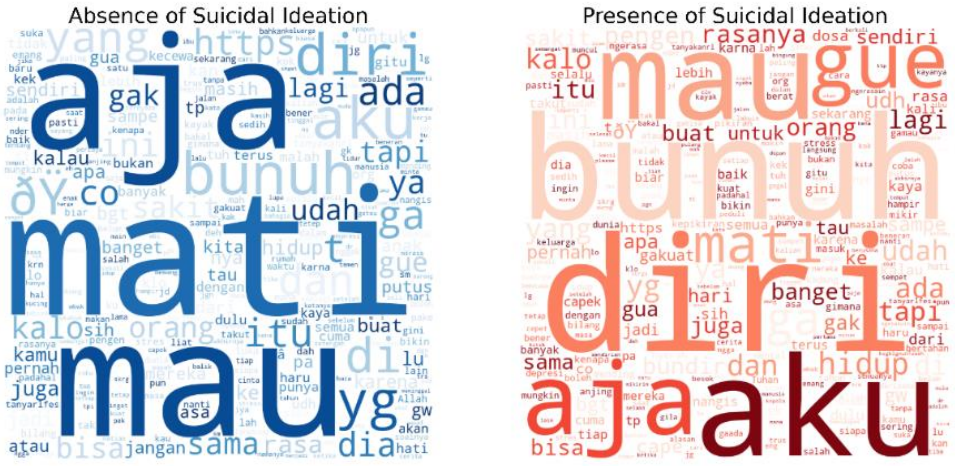
The total amount of textual data collected in this research is 10.128 tweets, with 1.967 (19,4%) tweets labeled as positive (presence of suicidal ideation) and 8.161 (80,6%) tweets labeled as negative (absence of suicidal ideation). Several text preprocessing methods are used, including case folding, stemming, the removal of mentions, hashtags, hyperlinks, punctuation, special characters, redundant whitespace, and stop words. An example of the processed text is shown in Table 4.

Table 4 Example of tweets before and after text preprocessing applications

Initial tweet	Tweets after text preprocessing	Label
Gile. 2hari ini otak gua gua pake buat mikirin trik bunuh diri yg kaga sakit wakakkaa dah stres akut ini keknya• »• •	gile 2hari otak gua gua pake buat mikirin trik bunuh diri yg kaga sakit wakakkaa dah stres akut keknya	1
Mentalku sudah hancur, tidak ada jalan keluar bagiku ketika ada masalah yg ada di pikiran ku hanya bunuh diri	mentalku hancur jalan keluar masalah yg pikir ku bunuh diri	1
Bila seseorang mati gantung diri, mustahil bila terdapat darah yg mengalir dari mulutnya kecuali ada luka tertentu. #ilmuDC	bila orang mati gantung diri mustahil bila darah yg alir mulut luka	0

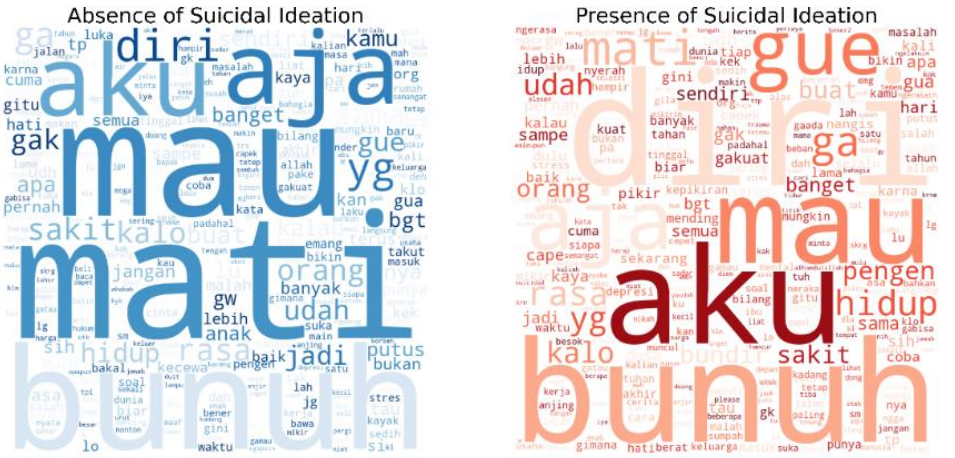
4.2 Data Exploration

A word cloud will be used to compare the appearance of words in different classes before and after text preprocessing has been done to the data. It can be seen in Figure 5 and Figure 6 that there is not much difference between the classes where suicidal ideation is present and where suicidal ideation is not present, even after the text processing steps have been done. This could happen because the dataset was collected using keywords associated with suicide and not just random tweets. Therefore, both classes have huge similarities in the appearance of words. Some common words that commonly appear in both classes are *aja*, *mati*, *mau*, *bunuh*, *aku*, and *diri*.



(a) (b)

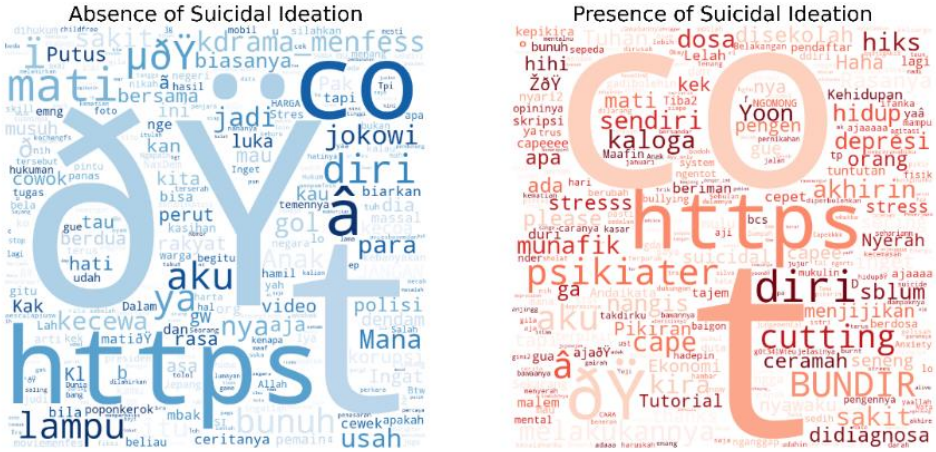
Figure 5 Word cloud comparison of the negative class (a) and positive class (b)



(a) (b)

Figure 6 Word cloud comparison after text preprocessing steps have been done of the negative class (a) and positive class (b)

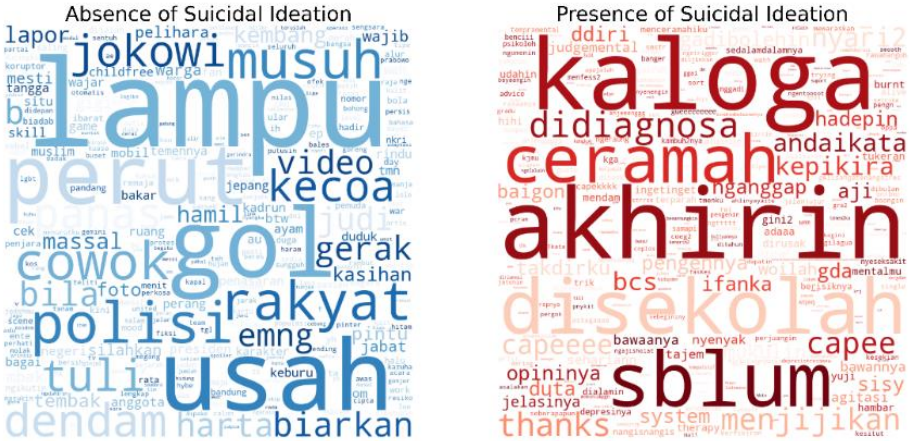
To handle this issue, another word cloud comparison is performed that only takes words in which the positive class and the negative class do not intersect. It can be seen in Figure 7, tweets that hadn't been processed contain numerous instances of special characters and texts that were hyperlink components (https, co, and t). In the class where suicidal ideation is absent, most of the words that appear are: 1) things unrelated to an actual suicide case but more on the sentence that also uses the term related to suicide in Indonesian (*bunuh diri* and *mati*); 2) discussing fictional works that are related to suicide cases or suicidal behavior; and 3) public discussion related to or talking about suicidal ideation. In the tweets containing suicidal ideation, most of the words that appeared were related to feelings of guilt (*dosa, munafik, maafin, menjijikan*), despair (*akhirin, nyerah*), loneliness (*sendiri*), or exhaustion (*cape, stress, depresi*). Other words that are closely related to suicide also appear frequently, such as the word *suicidal* itself; *cutting*, which is a term for a self-harm action of slitting one's own wrist; and *bundir*, which is a common abbreviation of *bunuh diri* (Indonesian word for suicide).



(a) (b)

Figure 7 Word cloud comparison without intersect word before text preprocessing of the negative class (a) and positive class (b)

After the text preprocessing has been done to the tweets as shown in Figure 8, the hyperlink components and the special characters have been cleaned. In the class where suicidal ideation is not present, most of the words that appeared are the words unrelated to actual suicide cases. The word *lampu* appears often because there is an Indonesian term '*mati lampu'* which actually means blackout. The word *gol* appears often because of the term *'gol bunuh diri'* which translates to 'own goal', a term in football when the players score a goal for the enemy team. The words *jokowi, rakyat, polisi, musuh, gerak,* and *massal* that appeared frequently are related to political topics that sometimes use words related to suicidal ideation like *langkah politik bunuh diri* (to describe an action by a politician that could actually end their career), *rakyat dibuat menderita* (to describe a political action done by the politician making the people suffer), and *pendukungnya bunuh diri massal* (to mock the supporters of a political party after losing their political campaign). Meanwhile, the words that appear in class with suicidal ideation present are words related to actual suicide cases like *akhirin* (wanting to end their own life) and *didiagnosa* (the user has been diagnosed with a physical/mental condition).



(a) (b)

Figure 8 Word cloud comparison without intersect word after text preprocessing of the negative class (a) and positive class (b)

4.3 Fasttext Word Embedding

The resulted embedding matrix from the pretrained fasttext embedding is shown in Table 5. The tokenized data will be transformed into vectors of dimension 300 to be fed as the inputs for the LSTM model. Each row in the matrix represents the n-gram characters while the columns represent their respective position in the embedding space. Each dimension (column) in the embedding matrix explains the relationships between words. Words with similar meanings or contexts tend to have similar vector representations, meaning they are closer to each other in the embedding space.

Table 5 The resulting embedding matrix from the pretrained fasttext

	0	1	2	...	298	299
0	0	0	0	...	0	0
1	-0,039	-0,042	-0,063	...	-0,022	0,169
2	0,006	-0,039	-0,188	...	-0,028	0,122
...	...	...	...	...	...	...
16793	-0,104	-0,154	0,057	...	0,131	0,026
16794	-0,042	0,006	0,062	...	0,009	0,05

4.4 LSTM Model Performance

Figure 9 shows how the whole process when the textual inputs are given up until the labels are predicted. The sentence is separated into words based on the whitespace, and then each word is represented by n-gram characters (this study uses 5-gram characters from the pretrained fasttext model). The 5-gram characters are assigned to a unique index (this process is called tokenization). Then, each input is transformed into a vector of the 5-gram characters token. Padding adds extra elements to the data sequences to ensure that all sequences have the same length. Each token in the inputs that has been represented by vectors of tokens is transformed to a vector of dimension 300 based on the embedding matrix that has been loaded. The numerical value of the vector representation of each 5-gram character is aggregated (summing their vector representation value) to represent the vector representation of the words. Finally, each word (represented by the vectors from the embedding matrix) in the textual inputs is used as an input (x_t) at the timestamp t and fed into each LSTM unit in the LSTM layer. Each word is processed sequentially in the order they appear in the sentence, and the LSTM units have memory to capture context from preceding words. The LSTM model processes the inputs and outputs the probability of a given input belonging to one of the two classes. If the probability is equal or higher than 0.5, then the model predicts it as 1 (positive case).

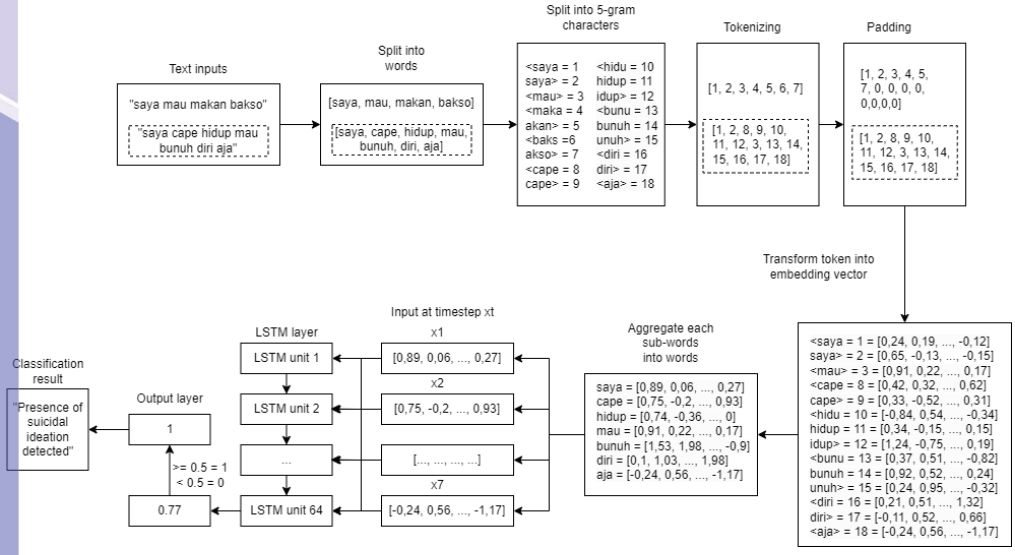


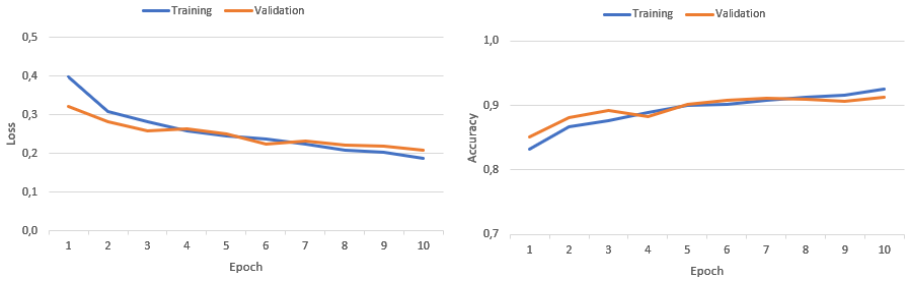
Figure 9 Simplified illustration of the model procedure

Table 6 displays the values of loss, accuracy, validation loss, and validation accuracy when no imbalanced class treatments are used and no text preprocessing techniques are applied. It can be seen that after each epoch, the value of loss in both the training and validation data is decreasing. Both the accuracy of the training and validation data also increases after each epoch, which indicates that the model can generalize well to new data.

Table 6 Loss and validation for each epoch

Epochs	Loss	Accuracy	Validation loss	Validation accuracy
1	0,3981	0,8321	0,3217	0,8514
2	0,3082	0,8666	0,2820	0,8815
3	0,2816	0,8763	0,2575	0,8924
4	0,2597	0,8890	0,2649	0,8830
5	0,2460	0,8993	0,2496	0,9023
6	0,2362	0,9015	0,2235	0,9082
7	0,2232	0,9080	0,2313	0,9116
8	0,2086	0,9134	0,2208	0,9102
9	0,2044	0,9153	0,2195	0,9062
10	0,1864	0,9247	0,2083	0,9126

Figure 10 explains that there is no overfitting in the LSTM model because the value of loss and accuracy of both the training and validation data are parallel. Overfitting happens when the loss of the training data decreases or the accuracy of the training data increases, but the reverse happen to the validation data. There is only a tiny gap between the loss and accuracy of training and validation data, which suggest that overfitting does not happen.



(a) (b)

Figure 10 Loss (a) and accuracy (b) of the training and validation data for each iteration of the epoch

Table 7 displays how different treatments to handle imbalanced class problem impact the model's performance using the data without any text preprocessing application. Without any imbalanced class treatment being done, the LSTM model had the best general performance by having 90% precision, 87% recall, and an 88% F1 score. The model without treatment also had decent sensitivity (78%) and high specificity (97%). By performing class weighting, the sensitivity of the model experienced an increase of 3% but also a tiny decrease in specificity (-3%), macro-average of precision (-4%), and macro-average of f1 score (-1%). By performing oversampling methods, the model sacrificed general performance for a notably better performance in sensitivity, especially when ADASYN (89%) and SMOTEENN (87%) are used compared to when no treatment is being done (78%). It can be concluded that by performing oversampling methods, the number of false negative (FN) cases that the classification model makes significantly decreased.

Table 7 Model performance comparison of each imbalanced class treatment

Treatment	Sensitivity	Specificity	Precision $_m$	Recall $_m$	F1 score $_m$
No treatment	0,78	**0,97**	**0,90**	**0,87**	**0,88**
Class weighting	0,81	0,94	0,86	0,87	0,87
ADASYN	**0,89**	0,77	0,72	0,83	0,74
SMOTEENN	0,87	0,7	0,68	0,78	0,68

Table 8 displays the confusion matrix to compare the model's performance without any imbalanced class treatment to the data treated by ADASYN. It can be clearly seen that performing ADASYN increased the instances of TP cases while consequently reducing the model's overall performance by making a lot more FP.

Table 8 Confusion matrix of the LSTM model when (a) no imbalanced class treatment were given and (b) treated by ADASYN

		Predicted	
		Negative	Positive
Actual	Negative	1577	57
	Positive	85	307

(a)

		Predicted	
		Negative	Positive
Actual	Negative	1257	377
	Positive	45	347

(b)

Table 9 shows the evaluation metrics of the LSTM model without imbalanced class treatment across different text preprocessing applications. The result shows that there is no significant increase in performance, both specifically and generally, after applying text preprocessing. Several text preprocessing methods even lead the model to decrease in general performance, notably when 1) removing mentions and hashtags, 2) removing punctuation, 3) removing special characters, and 4) all text preprocessing methods are applied. This could happen because the process of removing mentions, hashtags, punctuation, and special characters could lead to a loss of information that could have been captured by the fasttext word embedding.

Table 9 Model performance comparison for each preprocessing methods

Text preprocessing	Sensitivity	Specificity	Precision $_m$	Recall $_m$	F1 score $_m$
No text preprocessing	0,78	0,97	0,90	0,87	0,88
Case folding	0,80	0,95	0,87	0,87	0,87
Remove mentions and hashtags	0,71	0,97	0,89	0,84	0,86
Remove hyperlink	0,79	0,95	0,86	0,87	0,86
Remove punctuation	0,69	0,97	0,89	0,83	0,86
Remove special characters	0,68	0,97	0,89	0,83	0,85
Stop words removal	0,79	0,93	0,84	0,86	0,85
Stemming	0,84	0,94	0,86	0,89	0,87
Remove redundant whitespace	0,73	0,97	0,89	0,85	0,87
All text preprocessing	0,70	0,96	0,87	0,83	0,85

4.5 Model Deployment

The chosen model to be deployed is the one trained on data treated by ADASYN and without performing any text preprocessing methods. The classification model has been deployed on an online website and is publicly available at https://ipb.link/acn-model-sid-demo. People can try to submit their prompt and the model will predict whether the text contains suicidal ideation. Several test cases and their results are included in Table 10.

Table 10 Test cases after the model has been deployed

No	Test cases	Supposed label	Predicted label	Result
1	udah muak sama hidup mau akhirin semuanya aja	1	1	TP
2	mau mati aja lompat dari gedung tinggi	1	1	TP
3	wkwkkw semua orang gaada yang tau aja padahal gue sbnernya udh berapa kali attempt suicide	1	0	FN
4	Kucing saya mau bunuh diri	0	1	FP
5	LEBIH 1 TAHUN ADA RIBUAN MAHASISWA BUNUH DIRI #mahasiswa #mentalhealth Full Video Youtube: https://t.co/1aiJYVZxnP https://t.co/TdXynrY3se	0	0	TN

From the test cases shown in Table 10, the actual model performance when faced with real-world data is tested. The first and second examples show that the model could correctly predict (TP) the tweets where suicidal ideation is explicitly stated without the appearance of suicide words in Indonesia (*bunuh diri*). The third case shows that the model is unable to correctly detect (FN) suicidal ideation within the tweet. This could happen because the inputted prompt use both Indonesian and English, while the majority of the training data is written completely in Indonesian. In the fourth case, the model incorrectly predicted a tweet where suicidal ideation was not present as positive (FP). This could happen because the chosen model is the one that has better sensitivity but a worse general performance (resulting in more FP cases). Another possible reason is because the tweets in the training data were real tweets and it is highly unlikely that a user would tweet something similar to the prompt. The final test case is a tweet containing news related to suicide cases and the model correctly predicted it as having no suicidal ideation (TN).

V CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Among the 10.128 tweets crawled, 1.967 (19,4%) tweets exhibited signs of suicidal ideation were found. Tweets containing suicidal ideation tend to use words associated with negative emotions like sadness, loneliness, exhaustion, and hopelessness. In contrast, the remaining tweets lacking the presence of suicidal ideation consisted of tweets ranging from topics completely unrelated to suicide to discussion about suicidal ideation in fictional works, news or information related to suicide, and public discussion around the topics of suicidal ideation.

LSTM performed well in detecting tweets containing suicidal ideation when combined with fasttext word embedding. Without applying any text preprocessing methods and imbalanced class treatments, the model had outstanding performance with 78% sensitivity, 97% specificity, and an 88% F1 score. In this study case, text preprocessing applications are deemed unnecessary when fasttext word embedding is used. However, performing oversampling methods increased the ability to detect tweets containing suicidal ideation, which increases TP and decreases FN. The LSTM model trained on data treated by ADASYN resulted in a significant increase in sensitivity from 78% to 89% but also experienced a decrease in overall model performance.

5.2 Recommendation

In this study, performing text preprocessing has become unnecessary when fasttext word embedding is used. Nevertheless, it is evident that the deployed model remains excessively sensitive. Future research can investigate alternative methods for handling data with imbalanced classes that can enhance the model's sensitivity without compromising its specificity or F1 score. Another noteworthy constraint encountered in this research is the substantial cost associated with acquiring labeled data essential for training the model to identify tweets containing suicidal ideation. To overcome this challenge, upcoming investigations in unsupervised machine learning are crucial. These endeavors should be dedicated to devising strategies that mitigate the scarcity of the labeled data in this study case. Once the problem has been solved, future studies can delve into the utilization of advanced techniques such as Transformers or other state-of-the-art classification models to potentially obtain better performance in detecting texts containing suicidal ideation.

BIBLIOGRAPHY

[KEMENKES] Kementrian Kesehatan Republik Indonesia. 2021. KEMENKES BEBERKAN MASALAH PERMASALAHAN KESEHATAN JIWA DI INDONESIA. [accessed on 2023 Jan 29]. https://www.kemkes.go.id/article/print/21100700003/kemenkes-beberkan-masalah-permasalahan-kesehatan-jiwa-di-indonesia.html.

[UNSC] United Nations Statistical Commission. 2016. SDG Indicators: Official list of SDG indicators. [accessed on 2023 Feb 7]. https://unstats.un.org/sdgs/indicators/indicators-list/.

[WHO] World Health Organization. 2021. Suicide. [accessed on 2023 Jan 29]. https://www.who.int/news-room/fact-sheets/detail/suicide.

Adina N. 2020. Sentimen analisis multi-label pada ujaran kebencian dan umpatan di twitter Indonesia menggunakan pendekatan deep learning [skripsi]. Semarang: Universitas Negeri Semarang. [accessed on 2023 Jan 11]. http://lib.unnes.ac.id/.

Ahamed MA, Hasan KA, Monowar KF, Mashnoor N, Hossain MA. 2020. ECG heartbeat classification using ensemble of efficient machine learning approaches on imbalanced datasets. *2020 2nd International Conference on Advanced Information and Communication Technology, ICAICT 2020*.

Batista GEAPA, Prati RC, Monard MC. 2004. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*. 6(1):20–29. doi:10.1145/1007730.1007735.

Bittrich S, Kaden M, Leberecht C, Kaiser F, Villmann T, Labudde D. 2019. Application of an interpretable classification model on Early Folding Residues during protein folding 08 Information and Computing Sciences 0801 Artificial Intelligence and Image Processing. *BioData Min*. 12(1):1–16. doi:10.1186/S13040-018-0188-2/FIGURES/7.

Bojanowski P, Grave E, Joulin A, Mikolov T. 2017. Enriching Word Vectors with Subword Information. *Trans Assoc Comput Linguist*. 5:135–146. doi:10.1162/TACL_A_00051.

Chan K, Fang W. 2007. Use of the internet and traditional media among young people. *Young Consumers*. 8(4):244–256. doi:10.1108/17473610710838608/FULL/XML.

Chang HC. 2010. A new perspective on Twitter hashtag use: Diffusion of innovation theory. *Proceedings of the American Society for Information Science and Technology*. 47(1):1–4. doi:10.1002/MEET.14504701295.

Dalal MK, Zaveri MA. 2011. Automatic Text Classification: A Technical Review. *Int J Comput Appl*. 28(2):37–40. doi:10.5120/3358-4633.

Dalianis H. 2018. Evaluation Metrics and Evaluation. *Clinical Text Mining*. doi:10.1007/978-3-319-78503-5.

Dixon SJ. 2023 Jan 7. Twitter - Statistics & Facts | Statista. [accessed on 2023 Feb 7]. https://www.statista.com/topics/737/twitter/#topicOverview.

Elrahman SMA, Abraham A. 2013. A Review of Class Imbalance Problem. *Journal of Network and Innovative Computing*. 1:332–340. [accessed on 2023 Feb 2]. www.mirlabs.net/jnic/index.html.

Faadilah A. 2020 Jan 16. Analisis sentimen pada ulasan aplikasi tokopedia di google play store menggunakan metode long short term memory. [accessed on 2023 Jan 29]. https://repository.uinjkt.ac.id/dspace/handle/123456789/50432.

Feldman R, Sanger J. 2008. The Text Mining Handbook: Advanced Approaches to Analyzing Unstructured Data. *Computational Linguistics*. 34(1):125–127. doi:10.1162/COLI.2008.34.1.125.

Franklin JC, Ribeiro JD, Fox KR, Bentley KH, Kleiman EM, Huang X, Musacchio KM, Jaroszewski AC, Chang BP, Nock MK. 2017. Risk factors for suicidal thoughts and behaviors: A meta-analysis of 50 years of research. *Psychol Bull*. 143(2):187–232. doi:10.1037/BUL0000084.

Goldberg Y. 2017. *Neural Network Methods for Natural Language Processing*. Synthesis Lectures on Human Language Technologies. Cham: Springer International Publishing.

Grave E, Bojanowski P, Gupta P, Joulin A, Mikolov T. 2018 Feb 19. Learning Word Vectors for 157 Languages. *LREC 2018 - 11th International Conference on Language Resources and Evaluation*. [accessed on 2023 Feb 9]. https://arxiv.org/abs/1802.06893v2.

Han JH, Choi DJ, Park SU, Hong SK. 2020. Hyperparameter Optimization Using a Genetic Algorithm Considering Verification Time in a Convolutional Neural Network. *Journal of Electrical Engineering and Technology*. 15(2):721–726. doi:10.1007/S42835-020-00343-7/METRICS.

Harmer B, Lee S, Duong T vi H, Saadabadi A. 2020 Des 23. Suicidal Ideation. *Acute Medicine: A Symptom-Based Approach*.

He H, Bai Y, Garcia EA, Li S. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. *Proceedings of the International Joint Conference on Neural Networks*.

Hickman L, Thapa S, Tay L, Cao M, Srinivasan P. 2022. Text Preprocessing for Text Mining in Organizational Research: Review and Recommendations. *Organ Res Methods*. 25(1):114–146. doi:10.1177/1094428120971683/ASSET/IMAGES/LARGE/10.1177_1094428120971683-FIG1.JPEG.

Ho Y, Wookey S. 2020. The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling. *IEEE Access*. 8:4806–4813. doi:10.1109/ACCESS.2019.2962617.

Hochreiter S, Schmidhuber J. 1997. Long Short-Term Memory. *Neural Comput*. 9(8):1735–1780. doi:10.1162/NECO.1997.9.8.1735.

Khan L, Amjad A, Ashraf N, Chang HT, Gelbukh A. 2021. Urdu Sentiment Analysis with Deep Learning Methods. *IEEE Access*. 9:97803–97812. doi:10.1109/ACCESS.2021.3093078.

Kingma DP, Ba JL. 2014 Des 22. Adam: A Method for Stochastic Optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. [accessed on 2023 Jan 31]. https://arxiv.org/abs/1412.6980v9.

Klonsky ED, May AM, Saffer BY. 2016. Suicide, Suicide Attempts, and Suicidal Ideation. *https://doi.org/101146/annurev-clinpsy-021815-093204*. 12:307–330. doi:10.1146/ANNUREV-CLINPSY-021815-093204.

Kratzert F, Klotz D, Brenner C, Schulz K, Herrnegger M. 2018. Rainfall-runoff modelling using Long Short-Term Memory (LSTM) networks. *Hydrol Earth Syst Sci*. 22(11):6005–6022. doi:10.5194/HESS-22-6005-2018.

Lazaris A, Prasanna VK. 2021. An LSTM Framework for Software-Defined Measurement. *IEEE Transactions on Network and Service Management*. 18(1):855–869. doi:10.1109/TNSM.2020.3040157.

Luqyana WA, Cholissodin I, Perdana RS. 2018. Analisis Sentimen Cyberbullying pada Komentar Instagram dengan Metode Klasifikasi Support Vector Machine. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*. 2(11):4704–4713. [accessed on 2023 Feb 2]. https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/3051.

Nock MK, Borges G, Bromet EJ, Alonso J, Angermeyer M, Beautrais A, Bruffaerts R, Wai TC, De Girolamo G, Gluzman S, *et al.* 2008. Cross-national prevalence and risk factors for suicidal ideation, plans and attempts. *The British Journal of Psychiatry*. 192(2):98–105. doi:10.1192/BJP.BP.107.040113.

Park HW. 2003. Hyperlink Network Analysis: A New Method for the Study of Social Structure on the Web. *Connections*. 25(1):49–61.

Patel B, Shah D. 2013. Significance of stop word elimination in meta search engine. *2013 International Conference on Intelligent Systems and Signal Processing, ISSP 2013*.

Que J, Yuan K, Gong Y, Meng S, Bao Y, Lu L. 2020. Raising awareness of suicide prevention during the COVID-19 pandemic. *Neuropsychopharmacol Rep*. 40(4):392–395. doi:10.1002/NPR2.12141.

Rao A, Spasojevic N. 2016 Jul 8. Actionable and Political Text Classification using Word Embeddings and LSTM. [accessed on 2023 Feb 2]. https://arxiv.org/abs/1607.02501v2.

Riza MA, Charibaldi N. 2021. Emotion Detection in Twitter Social Media Using Long Short-Term Memory (LSTM) and Fast Text. *International Journal of Artificial Intelligence & Robotics (IJAIR)*. 3(1):15–26. doi:10.25139/IJAIR.V3I1.3827.

Robinson J, Cox G, Bailey E, Hetrick S, Rodrigues M, Fisher S, Herrman H. 2016. Social media and suicide prevention: a systematic review. *Early Interv Psychiatry*. 10(2):103–121. doi:10.1111/EIP.12229.

Ruby AU, Theerthagiri P, Jacob IJ, Vamsidhar Y. 2020. Binary cross entropy with deep learning technique for Image classification. *International Journal of Advanced Trends in Computer Science and Engineering*. 9(4):5393–5397. doi:10.30534/ijatcse/2020/175942020.

Sawhney R, Manchanda P, Singh R, Aggarwal S. 2018. A Computational approach to feature extraction for identification of suicidal ideation in tweets. *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Student Research Workshop*.

Sharma O. 2019 Feb 1. Deep Challenges Associated with Deep Learning. *Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Prespectives and Prospects, COMITCon 2019*.

Sir AY, Soepranoto AHH. 2022. Pendekatan Resampling Data Untuk Menangani Masalah Ketidakseimbangan Kelas. *J-ICON : Jurnal Komputer dan Informatika*. 10(1):31–38. doi:10.35508/JICON.V10I1.6554.

Srivastava N, Hinton G, Krizhevsky A, Salakhutdinov R. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*. 15:1929–1958. doi:10.5555/2627435.2670313.

Sun P, Liu P, Li Q, Liu C, Lu X, Hao R, Chen J. 2020. DL-IDS: Extracting features using CNN-LSTM hybrid network for intrusion detection system. *Security and Communication Networks*. 2020. doi:10.1155/2020/8890306.

Tiun S, Mokhtar UA, Bakar SH, Saad S. 2020. Classification of functional and non-functional requirement in software requirement using Word2vec and fast Text. *J Phys Conf Ser*. 1529(4):042077. doi:10.1088/1742-6596/1529/4/042077.

Tsai TH, Chang HT, Chang YC, Chang YS. 2017. Personality disclosure on social network sites. *Comput Human Behav*. 76:469–482. doi:10.1016/J.CHB.2017.08.003.

Utomo MS. 2013. Implementasi Stemmer Tala pada Aplikasi Berbasis Web. *Dinamik*. 18(1):41–45. doi:10.35315/DINAMIK.V18I1.1673.

Vijayarani S, Ilamathi M, Nithya. 2015. Preprocessing Techniques for Text Mining - An Overview. *International Journal of Computer Science & Communication Networks*. 5(1):7–16.

Wang JL, Jackson LA, Wang HZ, Gaskin J. 2015. Predicting Social Networking Site (SNS) use: Personality, attitudes, motivation and Internet self-efficacy. *Pers Individ Dif*. 80:119–124. doi:10.1016/J.PAID.2015.02.016.

Wang L, Littler T, Liu X. 2023. Hybrid AI model for power transformer assessment using imbalanced DGA datasets. *IET Renewable Power Generation*. 17(8):1912–1922. doi:10.1049/RPG2.12733.

APPENDICES

Appendix 1 Few examples of the dataset before and after text preprocessing and their respective label from each of the three annotators

Index	Initial tweet	Cleaned tweet	ACN	BME	AIP	Final Label
1	Gile. 2hari ini otak gua gua pake buat mikirin trik bunuh diri yg kaga sakit wakakkaa dah stres akut ini keknya• »• •	gile 2hari otak gua gua pake buat mikirin trik bunuh diri yg kaga sakit wakakkaa dah stres akut keknya	1	1	1	1
2	Mentalku sudah hancur, tidak ada jalan keluar bagiku ketika ada masalah yg ada di pikiran ku hanya bunuh diri	mentalku hancur jalan keluar masalah yg pikir ku bunuh diri	1	1	1	1
3	Bila seseorang mati gantung diri, mustahil bila terdapat darah yg mengalir dari mulutnya kecuali ada luka tertentu. #ilmuDC	bila orang mati gantung diri mustahil bila darah yg alir mulut luka	0	0	0	0
4	suasana batin ngerasa tertekan banget ðŸ˜”	suasana batin ngerasa tekan banget	0	0	0	0
5	@camareil KAMU MAU NGAPAIN WOI PLS JANGAN BUNUH DIRI	kamu mau ngapain woi pls jangan bunuh diri	0	0	0	0
6	Mau bunuh diri masih juga ngerepotin https://t.co/oJWZhJqWB8	mau bunuh diri ngerepotin	0	0	0	0
7	Udahlah mau gimana lagi ... Toh skrg lebih ke mati rasa kan. Cuma ngejalanin kewajiban aja yg bener takut dosa . Kalo dia mah ga tau deh masih bisa mikir apa enggak	udahlah mau gimana skrg lebih mati rasa kan cuma ngejalanin wajib aja yg bener takut dosa kalo mah ga tau deh mikir apa enggak	0	0	0	0

Index	Initial tweet	Cleaned tweet	ACN	BME	AIP	Final Label
8	@teaccakes sejujurnya mau pingsan aja enggak mau ngapa-ngapain tapi udah terlanjur nyemplung ðŸ˜žðŸ˜žðŸ˜ž deep down wanna take this chance as a networking chance instead but tetep anxious setengah mati	jujur mau pingsan aja enggak mau ngapangapain udah terlanjur nyemplung deep down wanna take this chance as a networking chance instead but tetep anxious tengah mati	0	0	0	0
9	@tubirfess plis bgt menfess2 kaya gini jujur bikin sedikit ke-trigger, bukan aku doang yg ngerasain sakit dan susahnya jd pengangguran dan udah usaha nyari kerja tp blm ada jodohnya. aku juga udah nganggur mau masuk 6bulan, yg keliatannya haha-hihi aja padahal nahan sedih setengah mati sampe sempet kepikiran untuk bunuh diri.	plis bgt menfess2 kaya gini jujur bikin sedikit ketrigger bukan aku doang yg ngerasain sakit susah jd anggur udah usaha nyari kerja tp blm jodohnya aku udah nganggur mau masuk 6bulan yg keliatannya hahahihi aja padahal nahan sedih tengah mati sampe sempet kepikiran bunuh diri	1	1	1	1
...	...	...	...	...	...	...
10.128	aku mau bunuh diri ah, gaada lagi hal baik yg tersisa buat aku	aku mau bunuh diri ah gaada baik yg sisa buat aku	1	1	1	1

Appendix 2 Confusion matrix for various imbalanced class treatment scenarios

1. No imbalanced class treatment – testing data

		Predicted	
		Negative (0)	Positive (1)
Actual	Negative (0)	1577	57
	Positive (1)	85	307

2. Class weighting – testing data

		Predicted	
		Negative (0)	Positive (1)
Actual	Negative (0)	1537	97
	Positive (1)	75	317

3. ADASYN – testing data

		Predicted	
		Negative (0)	Positive (1)
Actual	Negative (0)	1257	377
	Positive (1)	45	347

4. SMOTEENN – testing data

		Predicted	
		Negative (0)	Positive (1)
Actual	Negative (0)	1147	487
	Positive (1)	52	340

Appendix 3 Classification report for various imbalanced class treatment scenarios

1. No imbalanced class treatment – testing data

	precision	recall	f1-score	support
0	0,95	0,95	0,96	1634
1	0,84	0,78	0,81	392
accuracy			0,93	2026
macro avg	0,90	0,87	0,88	2026
weighted avg	0,93	0,93	0,93	2026

2. Class weighting – testing data

	precision	recall	f1-score	support
0	0,95	0,94	0,95	1634
1	0,77	0,81	0,79	392
accuracy			0,92	2026
macro avg	0,86	0,87	0,87	2026
weighted avg	0,92	0,92	0,92	2026

3. ADASYN – testing data

	precision	recall	f1-score	support
0	0,97	0,77	0,86	1634
1	0,48	0,89	0,62	392
accuracy			0,79	2026
macro avg	0,72	0,83	0,74	2026
weighted avg	0,87	0,79	0,81	2026

4. SMOTEENN – testing data

	precision	recall	f1-score	support
0	0,96	0,70	0,81	1634
1	0,41	0,87	0,56	392
accuracy			0,73	2026
macro avg	0,68	0,78	0,68	2026
weighted avg	0,85	0,73	0,76	2026

BIOGRAPHY

Alvin Christian Nataputra was born in Bogor on June 13, 2001. He is the second child of Irwan Semiadji Nataputra and Heni Hadianto. He graduated from an acceleration class at SMAN 11 Kabupaten Tangerang in 2018 and got accepted in the Statistics and Data Science program at IPB in 2019 through the Seleksi Bersama Masuk Perguruan Tinggi Negeri (SBMPTN) selection process.

During his time in the undergraduate program, Alvin actively sought out opportunities to grow and contribute. He joined AIESEC in IPB as Talent Management Intern (2019), and progressively assumed higher responsibilities, including roles as People Analytics (2020), Talent Analytics Manager (2021), and ultimately as Talent Strategist Team Leader (2021). Simultaneously, he was also an active member of the Data Analysis Department in Gamma Sigma Beta (2021). In the Department of Statistics, he contributes by taking initiatives in some department events by becoming the Head of Publication and Branding at Satria Data (2021) and Organizing Committee President at Welcoming Ceremony of Statistics (2022). In his pursuit of excellence, Alvin explored further avenues through the Kampus Merdeka program at Microsoft in the Data and Artificial Intelligence learning path. During his final year, he embarked on valuable internship experiences, serving as a Research Intern at BRI Research Institute and a Data Science Intern at Indosat Ooredoo Hutchison.