Pengaruh Metode Penyeimbangan Kelas Terhadap Tingkat Akurasi Analisis Sentimen pada *Tweets* Berbahasa Indonesia

http://dx.doi.org/10.28932/jutisi.v6i2.2743

Ivan Nathaniel Husada[#1], Hapnes Toba [✉*2]

[#] *Program Studi Magister Ilmu Komputer, Fakultas Teknologi Informasi*
Universitas Kristen Maranatha
Jl. Prof. drg. Surya Sumantri No.65 Bandung
[1]1879008@maranatha.ac.id
[2]hapnestoba@it.maranatha.edu

Abstract — **Nowadays internet access is getting easier to get. Because of the ease of access to the internet, almost all internet users have social media. Social media is widely used by users to call out their opinions or even to make complaints about a matter and also discuss a topic with other social media users. From many existing social media, one that is popularly used for that activity is Twitter. Sentiment analysis on Twitter has become possible because of the activities of these Twitter users. In this research, the authors explore sentiment analysis with bag-of-words and Term Frequency Inverse Document Frequency (TF-IDF) features extraction based on tweets from Indonesian Twitter users. The data obtained is in imbalanced condition, so that it requires a method to overcome them. The method for overcoming imbalanced *dataset* uses resampling approach which combine over and under sampling strategies. The results of sentiment analysis accuracies with Naïve Bayes and neural networks before and after input data resampling are also compared. Naïve Bayes methods that will be used are Multinomial Naïve Bayes and Complement Naïve Bayes, while the Neural Network architecture that will be used as a comparison are Recurrent Neural Networks, Long Short-Term Memory, Gated Recurrent Units, Convolutional Neural Networks, and a combination of Convolutional Neural Networks and Long Short-Term Memory. Our experiments show the following harmonic scores (F1) of the sentiment analysis models: the Multinomial Naïve Bayes F1 score is 55.48, Complement Naïve Bayes is 51.33, Recurrent Neural Network is 75.70, Long Short-Term Memory is 78.36, Gated Recurrent Unit is 77.96, Convolutional Neural Network is 76.12, and finally the combination of Convolutional Neural Networks and Long Short-Term Memory achieves 81.14.**

Keywords— **Imbalanced Data; Naïve Bayes; Neural Network; Sentiment Analysis; Twitter.**

I. PENDAHULUAN

Pada jaman sekarang ini masyarakat sudah dapat dengan mudah mendapatkan akses internet. Hal itu terjadi karena semakin banyaknya layanan *wifi* di tempat-tempat publik maupun semakin murahnya tarif internet yang disediakan oleh *provider*. Hal itu juga menyebabkan sebagian besar masyarakat memiliki akun media sosialnya masing-masing. Selain itu media sosial yang dapat digunakan oleh masyarakat pun semakin banyak jenisnya sehingga masyarakat dapat memilih untuk memakai media sosial mana yang dianggap lebih nyaman. Media sosial yang banyak digunakan masyarakat untuk menyerukan opininya atau pun melakukan protes terhadap suatu hal dan bertukar opini dengan pengguna media sosial lainnya terhadap suatu topik adalah media sosial *Twitter*. Karena hal tersebut maka analisis sentimen akan dapat dimanfaatkan untuk mencari tahu seperti apakah kecenderungan opini masyarakat yang diwakili oleh pengguna *Twitter* terhadap suatu topik tertentu.

Pada penelitian kali ini akan dilakukan percobaan melakukan analisis sentimen berbahasa Indonesia dengan kelas yang tidak seimbang pada *dataset* yang diambil dari media sosial *Twitter*. *Dataset* tersebut akan dipilih sehingga hanya data yang sesuai dengan konteks yang diinginkan yang berada dalam *dataset*. Setelah dipilih *dataset* akan dinormalisasi dan diberi label secara manual, oleh dua orang anotator, untuk menentukan nilai sentimen yang ada dari setiap *tweets*-nya. Algoritma klasifikasi yang dipakai pada penelitian analisis sentimen ini adalah algoritma *Multinomial Naïve Bayes* dan *Complement Naïve Bayes* dengan ekstraksi fitur *Bag-of-Words* dan *TF-IDF*. Selain itu juga akan digunakan beberapa arsitektur *Neural Network* seperti *Recurrent Neural Network*, *Long Short-Term Memory*, *Gated Recurrent Unit*, *Convolutional Neural Network*, dan kombinasi dari *Convolutional Neural Network* dan *Long Short-Term Memory* dengan ekstraksi fitur menggunakan *Word Embedding Vector FastText* sebagai pembanding.

Hal yang akan menjadi tujuan utama dalam penelitian adalah bagaimanakah perbandingan performa antara model dengan algoritma klasifikasi *Naïve Bayes* dengan arsitektur *Neural Network*. Hal khusus yang menjadi salah satu titik pengamatan dalam penelitian terkait dengan masalah ketidakseimbangan data pada kelas *dataset* sehingga dibutuhkan metode untuk menyeimbangkan kelas *dataset* tersebut.

Oleh karena itu pada penelitian ini juga akan dibandingkan efek metode menyeimbangkan data tersebut dengan hasil analisis sentimen sebelum data diseimbangkan. Dengan demikian akan dapat diketahui efek dari metode penyeimbangan data antara algoritma klasifikasi *Naïve Bayes* dan beberapa arsitektur *Neural Network*.

II. KAJIAN LITERATUR

Pada bagian ini akan dibahas beberapa teori yang berhubungan dengan penelitian yang dilakukan.

A. Twitter

Media sosial *Twitter* adalah media sosial *microblogging* yang memungkinkan penggunanya untuk mengirim atau membaca *tweets* yang dibatasi hingga 280 karakter sehingga dianggap pengguna *Twitter* akan dapat menyampaikan maksud dan tujuan dengan singkat, padat, dan jelas. Sehingga banyak pengguna *Twitter* menggunakan akun *Twitter*-nya masing-masing untuk mengutarakan pendapatnya terhadap suatu topik tertentu ataupun bertukar opini dengan pengguna *Twitter* lainnya [1]. Selain itu *Twitter* dipilih pada penelitian ini karena data pada *Twitter* lebih mudah untuk diakses dan terbuka untuk publik. Karena hal tersebut maka analisis sentimen akan dapat dimanfaatkan untuk mencari tahu seperti apakah kecenderungan opini masyarakat yang diwakili oleh pengguna *Twitter* terhadap suatu topik tertentu.

B. Analisis Sentimen

Analisis sentimen atau disebut juga *Opinion Mining* adalah bidang studi yang menganalisis opini, sentimen, evaluasi, penilaian, sikap, dan emosi orang terhadap entitas seperti produk, layanan, organisasi, individu, masalah, acara, topik, dan atributnya [2]. Analisis sentimen pada media sosial saat ini menjadi dapat dilakukan karena semakin banyaknya masyarakat yang menggunakan media sosial sebagai media untuk menyuarakan pendapat pribadi mereka.

C. Naïve Bayes

Naïve Bayes adalah teknik klasifikasi berbasis teorema bayes yang merupakan prinsip peluang statistika dengan mengkombinasikan pengetahuan sebelumnya dengan pengetahuan baru [3]. Algoritma klasifikasi *Naïve Bayes* dipilih dalam penelitian kali ini karena metode klasifikasi *Naïve Bayes* adalah algoritma klasifikasi yang berguna dalam penggunaan *dataset* yang besar, selain itu algoritma klasifikasi *Naïve Bayes* juga merupakan salah satu metode klasifikasi yang sederhana namun memiliki kemampuan dan akurasi tinggi [4]. Kekurangan dari algoritma klasifikasi *Naïve Bayes* adalah ketika data tidak seimbang maka algoritma klasifikasi *Naïve Bayes* akan memberikan pembobotan menjadi bias kepada kelas yang lebih banyak [5]. Pada penelitian ini algoritma klasifikasi *Naïve Bayes* yang akan dipakai adalah *Multinomial Naïve Bayes* dan *Complement Naïve Bayes.*

1) *Multinomial Naïve Bayes*: *Multinomial Naïve Bayes* adalah model pengembangan dari algoritma klasifikasi *Naïve Bayes* dimana kelas tidak hanya ditentukan dengan kata yang muncul, tapi juga dengan jumlah kemunculannya sehingga cocok untuk klasifikasi teks atau dokumen, terlebih lagi untuk ukuran dokumen yang besar [6].

2) Complement Naïve Bayes: Complement Naïve Bayes adalah model pengembangan dari algoritma klasifikasi *Naïve Bayes* yang cenderung memiliki performa yang lebih baik dalam klasifikasi ketika kelas dalam *dataset* tidak seimbang [7]. Perbedaan dengan *Multinomial Naïve Bayes* adalah *Complement Naïve Bayes* Menggunakan data *train* yang seimbang dari tiap kelas sehingga dapat mengurangi bias yang ada karena *dataset* tidak seimbang [5].

D. Neural Network

Neural Network adalah sistem model komputansi yang terinspirasi oleh jaringan neuron biologis, dimana neuron menghitung nilai *output* berdasarkan *input* [8]. *Neural Network* mengumpulkan pengetahuannya dengan mendeteksi pola dan hubungan dalam data [9]. *Neural Network* dianggap lebih baik karena *Neural Network* memiliki kemampuan untuk menemukan pola tersembunyi dalam data [10]. Pada penelitian ini model dari *Neural Network* yang digunakan adalah *Recurrent Neural Network, Long Short-Term Memory, Gated Recurrent Unit, Convolutional Neural Network*, dan kombinasi dari *Convolutional Neural Network* dan *Long Short-Term Memory*.

1) *Recurrent Neural Network*: *Recurrent Neural Network* adalah kelas *Neural Network* dimana koneksi antara unit membentuk siklus terarah sehingga menghasilkan keadaan internal dari *network* yang memungkinkannya untuk menunjukan perilaku dinamis yang sementara [11]. *Recurrent Neural Network* secara natural cocok untuk memproses *time series data* dan *sequential data* [12]. Pada klasifikasi teks, model berbasis *Recurrent Neural Network* melihat teks sebagai urutan kata, dengan maksud untuk menangkap dependensi kata dan struktur teks [10]. *Recurrent Neural Network* memiliki masalah *Vanishing Gradient* atau kondisi dimana *Long Term Component* bergerak secara eksponensial dengan cepat ke norma 0, sehingga mustahil untuk model untuk mempelajari korelasi antara kondisi yang bersifat sementara [13].

2) *Long Short-Term Memory*: *Long Short-Term Memory* adalah perkembangan dari model *Recurrent Neural Network* yang diperkenalkan untuk mengurangi masalah *Vanishing Gradient* [14]. *Long Short-Term Memory* dapat mengurangi masalah *Vanishing Gradient* dengan memperkenalkan *Memory Cell* untuk mengingat *Value* selama interval waktu, dan juga memperkenalkan tiga *Gate* yaitu *Input Gate, Output Gate*, dan *Forget Gate* untuk mengatur aliran informasi masuk dan keluar dari *Cell* sehingga dapat menangkap *Long-Term Dependencies* dengan lebih baik [10]. *Long Short-Term Memory* merupakan perkembangan model *Recurrent Neural Network* yang paling popular saat ini [12].

3) *Gated Recurrent Unit: Gated Recurrent Unit* diperkenalkan [15] untuk membuat setiap *Recurrent Unit* secara adaptif menangkap dependensi dari skala waktu yang berbeda [16]. *Gated Recurrent Unit* merupakan alternative yang lebih sederhana dibandingkan *Long Short-Term Memory* dan juga cukup popular [12]. Sama seperti *Long Short-Term Memory*, *Gated Recurrent Unit* juga dapat mengurangi masalah *Vanishing Gradient* dari *Recurrent Neural Network* [20].

4) *Convolutional Neural Network: Convolutional Neural Network adalah Feed Forward Neural Network* yang khusus digunakan untuk memproses data yang dapat disajikan secara terpisah [17]. Sebagian besar *Convolutional Neural Network* melibatkan tiga tahap yaitu: *Convolution Operation, Activation Function* seperti *Rectified Linear Activation Function* (*ReLU*) [18], dan *Pooling Function* seperti *Maxpooling* [19]. *Convolution Operation* rata-rata memiliki bobot atau memiliki *Smooth Estimation* dari *Windowed Input*. Salah satu kelebihan dari *Convolution Operation* adalah koneksi antara node dalam jaringan menjadi lebih jarang dengan mempelajari fitur yang kurang penting. Selain itu manfaat dari *Convolution Operation* adalah parameter sharing. *Convolutional Neural Network* membuat asumsi bahwa kernel yang dipelajari untuk satu posisi *input* dapat digunakan disetiap posisi. *Convolution Operator* sering kali meningkatkan kemampuan *learning* dari jaringan [20]. *Pooling Function* berfungsi menggantikan *output* dari spesifik *node* terdekat dengan kesimpulan statistik. Motivasi dibalik penambahan *Pooling Layer* adalah secara statistik melakukan *Down-Sampling* sejumlah fitur sehingga membuat representasi *invariant* dengan terjemahan kecil dari *input* dengan mempertahankan fitur penting [20]. Output akhir akan dihasilkan melalui *FullyConnected Layer* yang terletak setelah *Convolutional Layer* dan *Maxpooling* [20].

5) *CNN-LSTM: CNN-LSTM* adalah gabungan dari arsitektur *Convolutional Neural Network dan Long Short-Term Memory* untuk mendapatkan keuntungan dari kelebihan masing-masing arsitektur dengan cara menggunakan *Convolutional Neural Network* untuk mengekstrak urutan representasi dari frasa tingkat tinggi, dan dimasukkan ke dalam *Long Short-Term Memory* untuk mendapatkan representasi kalimat [21].

E. Metrik Penilaian

Pada penelitian kali ini metrik penilaian yang akan digunakan adalah Skor *F1*. Nilai akurasi tidak dipakai pada klasifikasi dengan *dataset* dengan kelas yang tidak seimbang karena nilai akurasi merupakan rasio prediksi kelas yang benar dari keseluruhan data sehingga hasil dari perhitungannya akan memberikan nilai sesuai dengan kelas mayoritas semata [22].

Nilai Skor *F1* akan dijadikan acuan penilaian karena nilai *F1* melakukan perhitungan dengan menggunakan bobot dari nilai presisi dan *recall*. Dimana nilai presisi adalah rasio perhitungan kelas yang tepat dari semua kelas yang diprediksi sehingga cocok untuk dijadikan nilai akurasi dari kelas minoritas, sedangkan *recall* adalah rasio prediksi kelas yang benar dibandingkan keseluruhan data yang benar. Kombinasi perhitungan yang harmonis terhadap hasil presisi dan *recall* dengan rumus yang dapat dilihat pada formula nomor 1:

$$\text{Skor } F1 = \frac{2 * (Recall * Presisi)}{(Recall + Presisi)} \qquad (1)$$

Rumus tersebut akan menghasilkan skor *F1* dan sangat representatif sebagai metrik evaluasi untuk *dataset* dengan kelas yang tidak seimbang [22].

III. SUMBER DATA DAN METODOLOGI

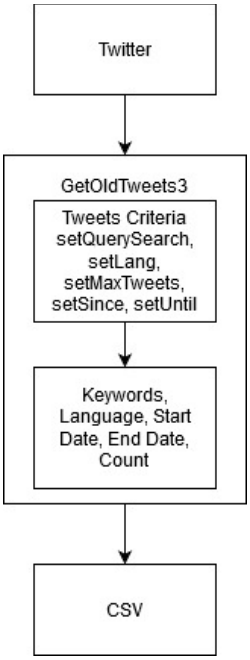
Sumber Data yang digunakan berasal dari media sosial *Twitter*, dan menggunakan *library python GetOldTweets3* untuk melakukan pengambilan data. Data yang diambil dengan menggunakan *library GetOldTweets3* adalah data dengan kriteria pengambilan data *Text Query* atau teks *Tweets*, *Language, Max Tweets*, *Since*, dan *Until*. Dengan parameter *keywords* sebagai kata kunci untuk topik pengambilan data, *Start Date* sebagai tanggal awal pengambilan data, *End Date* sebagai batasan tanggal akhir pengambilan data, dan *Count* yang berfungsi untuk menentukan jumlah data yang akan diambil dari *Twitter*, dimana jika nilai untuk *Count* diberikan menjadi 0 maka akan mengambil seluruh data yang memungkinkan dari kata kunci dengan ketentuan jeda waktu *Tweets* antara nilai *Start Date* dan *End Date* [23]. Hasil pengambilan data akan disimpan menjadi *format CSV*. Gambar 1 adalah diagram alur pengambilan data di Twitter dengan *library GetOldTweets3.*

A. Twitter

Kata kunci kunci yang menjadi topik penelitian adalah ‘Maranatha’, ‘marnath’, dan ‘marnat’ dengan konteks yang diinginkan berhubungan dengan Universitas Kristen Maranatha. Berdasarkan data yang diambil dengan jangka waktu 1 januari 2018 hingga 31 desember 2019 didapatkan data sebanyak 17.304 *Tweets. Tweets* yang telah didapat tersebut akan dipilih secara manual agar semua *Tweets* yang ada adalah *Tweets* yang sesuai dengan konteks kata kunci

yang diinginkan. Jumlah *Tweets* setelah dipilih adalah 5.367 *Tweets*. Setelah *Tweets* dipilih maka kata-kata yang tidak baku dan kata-kata dari bahasa daerah dinormalisasi, sehingga kata-kata dengan konteks yang sama akan menjadi satu kata.

Setelah itu *Tweets* yang didapat akan diberi label secara manual 1 yang berarti positif, 0 yang berarti netral, dan -1 yang berarti negatif. Setelah pemberian label maka didapat data positif sebanyak 970 data, data netral sebanyak 3.907 data, dan data negatif sebanyak 491 data. Berdasarkan data tersebut dapat disimpulkan data yang digunakan pada penelitian ini merupakan data yang tidak seimbang sehingga dibutuhkan metode *resampling* untuk mengatasinya.



Gambar 1. Diagram alur pengambilan data

B. GetOldTweets3

GetOldTweets3 adalah *library* khusus *python3* yang digunakan untuk mengakses *Tweets* lama [23]. *Library GetOldTweets3* dipilih sebagai alat untuk pengambilan *Tweets* lama di *Twitter* karena *Twitter Official API* memiliki batasan waktu dalam pengunaannya. *Twitter Official API* tidak menyediakan akses ke *Tweets* lama yang lebih lama dari seminggu. Selain itu *Twitter Official API* memilik batas pengambilan 180 data per 15 menit [24]. Cara kerja *library GetOldTweets3* adalah dengan meniru fungsi *search Twitter* pada *browser* dengan cara memasukkan kata kunci dan melakukan *scroll* ke bawah untuk mendapatkan data yang di-*post* lebih terdahulu [23].

Pengaturan yang dilakukan *pada library GetOldTweets3* adalah dengan kata kunci 'Maranatha', 'marnath', dan 'marnat'. Selanjutnya adalah dengan mengatur pembatasan pengambilan data yaitu dengan *Start Date* tanggal 2018-01-01 dan *End Date* tanggal 2019-12-31 dengan *Count* diberi nilai 0 untuk pengambilan semua data yang ada dari jeda waktu tersebut. Data yang diambil adalah data yang berbahasa Indonesia saja, teks *Tweets* dan tanggal *posting Tweets* dan hasilnya akan disimpan dalam *format CSV*.

C. Data Cleaning

Proses *Data Cleaning* adalah proses untuk mengubah data yang tidak terstruktur menjadi data yang terstruktur [25]. Hasil contoh dari proses data tweets awal, data yang dinormalisasi, dan data yang sudah mengalami proses data cleaning akan dapat dilihat di Tabel I. Proses *data cleaning* yang dilakukan adalah:

1) *Case Folding: Case Folding* adalah proses mengubah semua data teks menjadi huruf kecil.

2) Cleansing: Cleansing adalah menghapus data yang dianggap *noise*. Dalam penelitian ini data yang dihapus adalah tanda baca, angka, teks dengan pattern tertawa yaitu haha, hehe, hihi, wkwk, dan kwkw, spasi berlebih, *mention username*, dan hapus *link* yang ada pada data *Tweets*. Selain itu dilakukan juga penghapusan *stopwords* dengan menggunakan *library PySastrawi*. *Stopwords* adalah adalah kata-kata tidak berarti yang memiliki kekuatan diskriminasi rendah [26]. *Stopwords* sering dihilangkan pada tahapan *preprocessing* karena dianggap memiliki nilai informasi yang rendah [27].

3) Stemming: Stemming adalah mereduksi kata menjadi bentuk akarnya atau akar semantiknya [28]. *Stemming* adalah alat *preprocessing* teks yang sering digunakan untuk *text retrieval* yang efisien [29], *machine translation* [30], *document summarization* [31] dan *text classification* [32].

TABEL I
CONTOH PROSES DATA *CLEANING* PADA *TWEETS*

***Tweets* Awal**	**Setelah Normalisasi**	**Setelah Normalisasi dan *Cleaning***	**Nilai Sentimen**
Saya mainnya ke daerah marnath doang pak	Saya mainnya ke daerah marnath saja pak	main daerah marnath pak	0
KampusQ maranatha yg kaya mall	Kampus saya maranatha yang seperti mall	kampus maranatha seperti mall	1
Sekolah di Bintaro, global jaya. Urg ge di marnath basa eta ngan 80rb / sks	Sekolah di Bintaro, global jaya. saya juga di marnath waktu itu hanya 80 ribu / sks	sekolah bintaro global jaya juga marnath waktu hanya ribu sks	0
Sarua euy hanjakal oge ningali umak nyanyi teknik industri ada di maranatha pas ospek	Sama menyesal juga melihat umak nyanyi teknik industri ada di maranatha pas ospek	sama sesal lihat umak nyanyi teknik industri di maranatha pas ospek	-1
DI DEKET MARNAT	DI DEKAT MARNAT	dekat marnat bagus sekali	1

***Tweets* Awal**	**Setelah Normalisasi**	**Setelah Normalisasi dan *Cleaning***	**Nilai Sentimen**
HAHXHSHSHSHS BAGUS BGT TERUS MURAH	BAGUS SEKALI DAN MURAH	murah	
That's why dari dulu saya pengin bgt FK Marnat atau Atma Jaya. Mereka lebih jauh menerapkan toleransi. Apa daya engga boleh. Bisa ngerasain bgt bedanya sama di kampusku yg sekarang	Karena itu dari dulu saya ingin sekali fakultas kedokteran marnat atau AtmaJaya. Mereka lebih jauh terapkan toleransi. Apa daya tidak boleh. Bisa rasakan sekali beda dengan di kampus aku yang sekarang.	itu dulu ingin sekali fakultas dokter marnat atmajaya lebih jauh terap toleransi apa daya boleh rasa sekali beda di kampus aku sekarang	1

D. PySastrawi

PySastrawi atau *python* sastrawi adalah *library python* yang berfungsi untuk mengubah kata berbahasa Indonesia menjadi bentuk dasarnya atau *stemming* [33]. *PySastrawi* adalah *port python* dari proyek asli sastrawi yang ditulis dalam PHP [33]. Sastrawi menerapkan algoritma Nazief dan Adriani dalam proses *stemming*-nya. Proses *stemming* dari sastrawi sangat bergantung pada kamus kata dasar [34].

E. Feature Extraction

Feature Extraction atau ektraksi fitur pada klasifikasi teks adalah proses mengeluarkan daftar kata-kata dari data teks dan kemudian mengubahnya menjadi fitur yang dapat digunakan oleh algoritma klasifikasi [35]. Pada penelitian ini ekstraksi fitur yang akan digunakan adalah *Bag-of-Words* dan *TF-IDF* untuk algoritma klasifikasi *Multinomial Naïve Bayes* dan *Complement Naïve Bayes*, dan juga *Pre-Trained Word Embedding Vector* dari *FastText* untuk *Neural Network*.

1) Bag-of-Words: Bag-of-Words adalah metode ekstraksi fitur yang membentuk fitur kehadiran dari kata pada dokumen yang digunakan sehingga dapat menentukan frekuensi kata yang ada dalam dokumen [35]. Hasil dari penyimpanan frekuensi akan berbentuk vektor. Kekurangan dari metode ini adalah kata-kata dengan frekuensi yang lebih tinggi akan menjadi dominan dalam data dimana ada kemungkinan kata-kata ini tidak memberikan banyak informasi untuk model sehingga kata-kata spesifik yang kemungkinan memberikan lebih banyak informasi menjadi diabaikan [35].

2) TF-IDF: TF-IDF atau *Term Frequency Inverse Document Frequency* adalah metode ekstraksi fitur yang mengatasi kekurangan dari metode *Bag-of-Words* dimana frekuensi kata-kata dihitung ulang dengan mempertimbangkan seberapa sering kata-kata tersebut muncul di semua dokumen [35]. *TF* atau *Term Frequency* adalah frekuensi kata-kata yang ada pada dokumen, dan *IDF* atau *Inverse Document Frequency* adalah skor kata-kata dari semua dokumen. Skor ini dapat menyorot kata-kata yang unik dan kata-kata yang mewakili informasi yang diperlukan dalam dokumen tertentu [35].

3) Word Embedding Vektor FastText: FastText adalah *library* yang dikembangkan oleh *FaceBook* untuk klasifikasi dan representasi teks dimana *FastText* mengubah teks menjadi vektor berkelanjutan yang dapat digunakan pada tugas terakit bahasa apapun [36]. *FastText* berkerja dengan cara mengekstrak semua kata dengan *n-gram* untuk mempelajari representasi dari setiap kata [37]. Penelitian ini akan menggunakan *Pre-Trained Word Vectors Multi-Lingual* dengan Bahasa Indonesia dari *FastText* [36].

F. Cross Validation StratifiedKFold

Cross Validation StratifiedKFold adalah *Cross Validation* dengan metode *stratification*. *Stratification* adalah pendekatan untuk mempertahankan proporsi kelas yang asli dalam *subsets* yang dihasilkan [38]. *Stratified K-Fold* adalah pendekatan validasi yang cocok untuk *dataset* tidak seimbang [39] sehingga cocok digunakan untuk penelitian ini karena data yang digunakan pada penelitian ini merupakan data dengan distribusi kelas yang tidak seimbang.

G. Imbalanced Data

Imbalanced Data atau data dengan kelas tidak seimbang yang akan digunakan pada penelitian ini adalah data yang berjumlah 5367 data dan dibagi menjadi 970 data kelas positif, 3907 data kelas netral, dan 491 data kelas negatif. Tabel II adalah contoh data Tweets dengan kelas sentimennya.

TABEL III
CONTOH KELAS SENTIMEN PADA DATA *TWEETS*

Tweet	***Sentiment***
Maranatha mahal ya. .	Negatif
jurusan IT maranatha kak? aku SI kwkwkw	Netral
Maranatha terbaik emang	Positif
marnat itu mahal	Negatif
marnat sudah libur kok	Netral
ingin pindah ke marnat ternyata besar sekali dan bagus sekali	Positif
Marnath seram kalau malam begini	Negatif
marnath kalau tidak salah masih buka	Netral
marnath kak, insyaallah yang terbaik deh :')	Positif

Berdasarkan jumlah data tersebut maka dapat disimpulkan data yang dimiliki adalah data dengan kelas yang tidak seimbang. Untuk mengatasi ketidakseimbangan dalam kelas data dibutuhkan metode *resampling* untuk menyeimbangkan data. Metode *resampling* dapat berupa *Over Sampling, Under Sampling*, dan kombinasi keduanya. Pada penelitian kali ini metode *resampling* data yang akan digunakan untuk mengatasi data dengan kelas yang tidak seimbang adalah metode gabungan dari *over sampling* dan

under sampling yaitu *SMOTETomek*. Dan hasil *resampling* akan dilihat efeknya pada algoritma klasifikasi dan arsitektur *Neural Network* yang dipilih.

1) *SMOTE: SMOTE* atau *Synthetic Minority Over-sampling Technique* adalah metode *Over Sampling* dengan menambahkan data kelas minoritas dengan membuat data sintetis terhadap kelas minoritas [40].

2) *Tomek Links*: *Tomek Links* adalah metode *Under Sampling* dengan menggunakan *Nearest-Neighbour Rule* dimana mendeteksi 2 kelas berbeda yang berdekatan. Dan menghapus data dari kelas mayoritas [41].

3) *SMOTETomek: SMOTETomek* adalah metode gabungan antara *Over Sampling* dan *Under Sampling*. Dimana data kelas minoritas ditambahkan dengan metode *SMOTE*, dan data kelas mayoritas yang berdekatan dengan kelas minoritas akan dikurangi dengan *TomekLinks* sehingga distribusi kelas yang diinginkan tercapai [39]. Strategi yang digunakan untuk *resampling* data menggunakan *SMOTETomek* pada penelitian kali ini adalah melakukan *SMOTE* dengan *sampling strategy not majority*, dan *TomekLinks* dengan *sampling strategy majority*. Hasil dari *resampling* pada data train dengan menggunakan SMOTETomek akan dapat dilihat pada Gambar 2.

```
Hasil SMOTETomek pada train data label positif: 2929
Hasil SMOTETomek pada train data label neutral: 2929
Hasil SMOTETomek pada train data label negatif: 2929
```

Gambar 2. Hasil *resampling* pada data *train*

IV. RANCANGAN SISTEM

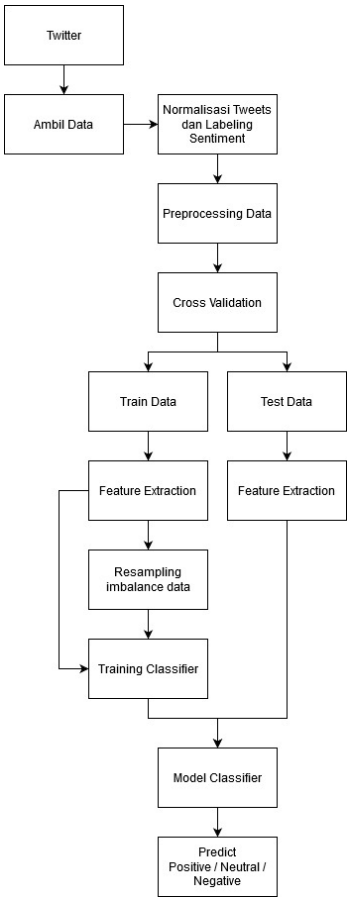
Bagian ini akan menjelaskan rancangan sistem dari analisis sentimen dengan menggunakan metode algoritma klasifikasi *Naïve Bayes* dan *Neural Network*.

A. Naïve Bayes

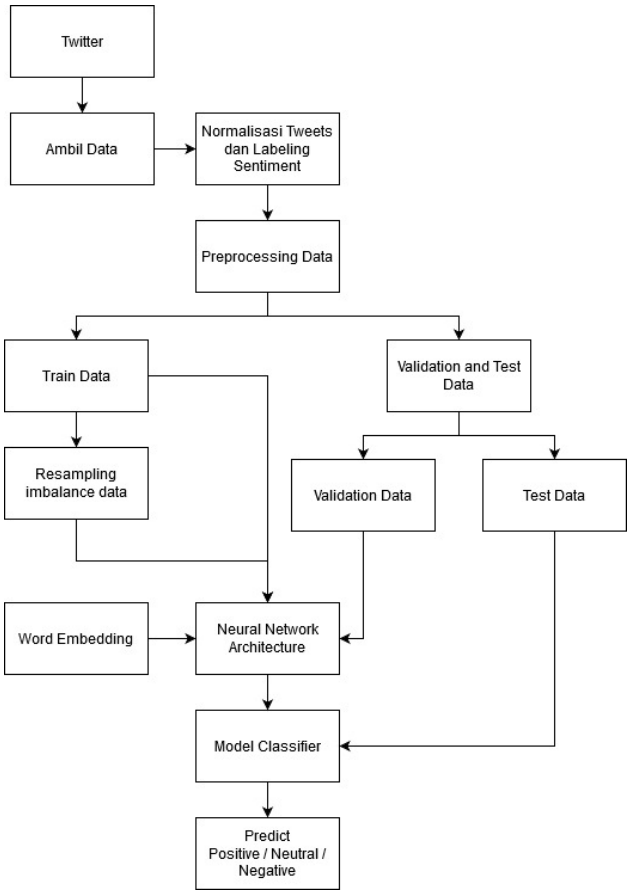
Data yang terdapat pada media sosial *Twitter* akan diambil berdasarkan kata kunci sebagai topik dengan menggunakan *library GetOldTweets3* dengan *Tweets* kriteria adalah *Tweets* berbahasa Indonesia saja yang terdapat pada rentang waktu 1 Januari 2018 dan 31 Desember 2019. Data yang diambil akan disimpan dalam bentuk dokumen *CSV*. Data tersebut akan dinormalisasi secara manual. Tahapan normalisasi data yang dilakukan adalah dengan memilih *Tweets* yang didapat agar *dataset* hanya berisi kata kunci dengan topik yang diinginkan saja, mengubah kata-kata dengan sinonim berbeda tetapi memiliki konteks yang sama menjadi satu kata, menerjemahkan bahasa daerah menjadi Bahasa Indonesia tanpa merubah konteksnya, mengubah singkatan-singkatan agar menjadi bentuk aslinya, menghapus *username*, dan *hashtag* pada data *Tweets*, dan menyatukan nama tempat atau jalan yang terdiri lebih dari satu kata menjadi satu kata saja. Setelah tahapan normalisasi data *Tweets* akan di beri label sentimen secara manual dimana 1 berarti positif, 0 berarti netral, dan -1 berarti negatif. Setelah itu data akan di-*preprocessing* atau dibersihkan dengan cara mengubah data menjadi undercase, menghapus tanda baca, angka, spasi berlebih, link, dan teks dengan konteks tertawa dengan pola haha, hehe, hihi, dan wkwk. Kemudian *preprocessing* selanjutnya adalah menghapus *stop words* dari data *Tweets*, dan melakukan *stemming* atau mengubah kata-kata pada data menjadi bentuk dasarnya. Setelah proses *preprocessing* selesai, data akan di bagi dengan menggunakan metode *Cross Validation StratifiedKFold* menjadi 2 yaitu data *train* dan data *test*.

Setelah itu data *train* dan data tes akan diubah menjadi bentuk vektor pada bagian ekstraksi fitur. Ekstraksi fitur yang dipilih adalah metode *Bag-of-Words* dan *TF-IDF*. Setelah data di ubah menjadi vektor maka data *train* akan di *resampling* dengan menggunakan metode *SMOTETomek* agar data yang tidak seimbang dapat menjadi data dengan distribusi yang seimbang. Setelah data *train* memiliki distribusi kelas yang seimbang, data *train* akan diaplikasikan menggunakan algoritma klasifikasi *Multinomial Naïve Bayes* dan *Complement Naïve Bayes*. Selain itu data *train* awal dengan bentuk vektor hasil ekstraksi fitur tanpa *resampling* juga akan langsung diaplikasikan ke algoritma klasifikasi sebagai pembanding dengan hasil *resampling*. Hasil penilian algoritma klasifikasi adalah nilai dari skor *F1*. Gambar 3 memberikan rancangan sistem dari algoritma klasifikasi *Naïve Bayes*.

p-ISSN : 2443-2210
e-ISSN : 2443-2229



Gambar 3. Rancangan sistem algoritma klasifikasi *Naïve Bayes*



Gambar 4. Rncangan sistem *Neural Network*

B. Neural Network

Rancangan sistem pada metode *Neural Network* dari proses pengambilan data hingga *preprocessing* data sama dengan yang dilakukan pada metode *Naïve Bayes*, tapi pada *Neural Network* proses pembagian *dataset* akan dibagi menjadi 3 *dataset*, yaitu data *train*, data *validation*, dan data *test*. Data *train* akan digunakan untuk data *training*, sementara data *validation* akan digunakan sebagai data untuk validasi arsitektur *Neural Network* dan test *data* adalah data yang akan digunakan sebagai data *test* pada model *Neural Network*. Setelah proses pembagian *dataset* maka data *train* akan di *resampling* agar distribusi kelasnya seimbang dengan menggunakan metode *SMOTETomek*.

Penelitian ini menggunakan *Pre-Trained Word Embedding Vector* dari *FastText* yang akan digunakan pada arsitektur *Neural Network* sebagai ekstraksi fitur. Selain itu data *train* yang tidak mengalami *resampling* juga akan diaplikasikan ke arsitektur *Neural Network* sebagai pembanding. Hasil arsitektur *Neural Network* akan berupa nilai skor *F1*. Gambar 4 merupakan Gambaran rancangan sistem dari arsitektur *Neural Network*.

V. IMPLEMENTASI SISTEM

Penelitian ini akan membandingkan hasil dari model *training* menggunakan metode klasifikasi *Multinomial Naïve Bayes* dan *Complement Naïve Bayes*. Selain itu metode *Neural Network* juga akan dicoba sebagai pembanding. Selain itu hasil perbandingan dari *input* yang sudah di-*resampling* dengan metode *SMOTETomek* akan dibandingkan dengan hasil dari *input* yang tidak di-*resampling*.

Pada algoritma klasifikasi *Naïve Bayes* akan menggunakan ekstraksi fitur *Bag-of-Words* dan TF-IDF yang sudah di jadikan bentuk vektor dengan *library sckit-learn*, dan juga *Cross Validation* dengan metode *StratifiedKFold* dengan *library sckit-learn*. Sedangkan pada *Neural Network* ekstraksi fitur kan menggunakan *Pre-Trained Word Embedding Vector* dari *FastText*.

Hasil model arsitektur *Neural Network* juga akan dibandingkan dengan *input* sebelum *resampling* dan hasil dengan *input* setelah *resampling* dengan *SMOTETomek*. Proses *resampling* data akan dilakukan pada data *train* setelah *dataset* dibagi antara data *train*, data *validation*, dan data *test*.

A. Multinomial Naïve Bayes

Metode yang digunakan pada algoritma klasifikasi *Multinomial Naïve Bayes* adalah dengan ekstraksi fitur *Bag-of-Words* dan *TF-IDF*, dan resampling data dengan metode *SMOTETomek*.

1) *Bag-of-Words*: Pada ekstraksi fitur *Bag-of-Words* akan menggunakan *CountVectorizer* dari *library sckit-learn* untuk mengubah hasil perhitungan frekuensi antar kata menjadi bentuk vektor, dan dengan nilai *n-gram* dari 1 sampai 3. Pada parameter untuk *resampling* data, *SMOTE* menggunakan *resampling strategy not majority* untuk menambahkan data pada semua jumlah kelas yang lebih sedikit hingga memiliki jumlah yang sama, dan pada *TomekLinks* menggunakan *sampling strategy majority* untuk membersihkan data kelas mayoritas yang berdekatan dengan kelas minoritas. *Random state* pada setiap pada setiap parameter diberikan nilai 5. *Setting* parameter yang digunakan pada percobaan dapat dilihat pada Tabel III.

2) TF-IDF: Pada ekstraksi fitur *TF-IDF* akan menggunakan *TFIDFVectorizer* dari *library sckit-learn*. *Setting* parameter yang digunakan adalah *ngram range* 1 sampai 3. Pada parameter untuk *resampling* data, *SMOTE* menggunakan *resampling strategy not majority* untuk menambahkan data pada semua jumlah kelas yang lebih sedikit hingga memiliki jumlah yang sama, dan pada *TomekLinks* menggunakan *sampling strategy majority* untuk membersihkan data kelas mayoritas yang berdekatan dengan kelas minoritas. *Random state* pada setiap pada setiap parameter diberikan nilai 5. *Setting* parameter yang digunakan pada percobaan dapat dilihat pada Tabel III.

TABEL IIIII
SETTING PARAMETER ALGORITMA KLASIFIKASI NAÏVE BAYES

Algoritma	Parameter	Deskripsi	Nilai
MultinomialNB, ComplementNB	alpha	*smoothing* untuk zero probability	1
	fit_prior	pelajari probabilitas dari kelas prior	True
CountVectorizer, TFIDFVectorizer	max_features	batas maksimal *vocabulary*	None
	ngram_range	*range ngram* untuk kata yang berbeda	1, 3
SMOTE	random_state	*randomization* menggunakan *random number generator*	5
	sampling_strategy	menentukan kelas target *resampling*	not majority
TomekLinks	random_state	*randomization* menggunakan *random number generator*	5
	sampling_strategy	menentukan kelas target *resampling*	majority
SMOTETomek	random_state	*randomization* menggunakan *random number generator*	5

B. Complement Naïve Bayes

Metode yang digunakan pada algoritma klasifikasi *Complement Naïve Bayes* adalah dengan ekstraksi fitur *Bag-of-Words* dan *TF-IDF*, dan resampling data dengan metode *SMOTETomek*.

1) *Bag-of-Words*: *Setting* parameter yang digunakan pada ekstraksi fitur *Bag-of-Words* pada algoritma klasifikasi *Complement Naïve Bayes* sama dengan *setting* parameter yang dilakukan pada ekstraksi fitur *Bag-of-Words* pada algoritma klasifikasi *Multinomial Naïve Bayes*. *Setting* parameter yang digunakan pada percobaan dapat dilihat pada Tabel III.

2) TF-IDF: Setting parameter yang digunakan pada ekstraksi fitur *TF-IDF* pada algoritma klasifikasi *Complement Naïve Bayes* sama dengan *setting* parameter yang dilakukan pada ekstraksi fitur TF-IDF pada algoritma klasifikasi *Multinomial Naïve Bayes*. *Setting* parameter yang digunakan pada percobaan dapat dilihat pada Tabel III.

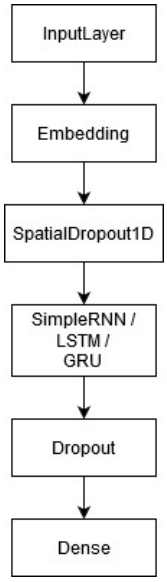
C. Neural Network

Setting parameter yang dilakukan pada semua arsitektur *Neural Network* adalah pada *Input Layer* diberikan nilai maksimal panjang data *Tweets* yang ada pada *dataset*, nilai *input* pada *Embedding Layer* adalah nilai panjang maksimal dari data *Tweets* yang ada pada *dataset* dan ukuran dimensi dari *Pre-Trained Word Embedding Vector* dari *FastText*, *Dropout* dengan nilai 0.5 dimana *Dropout* pada perecobaan kali ini digunakan sebagai *Hidden Layer* sebelum *Output Layer*, dan *Dense* pada *Output Layer* diberi *input* sesuai dengan jumlah kelas yang akan diprediksi. *Batch Size* pada semua arsitektur *Neural Network* diberikan nilai 12 dan *epoch* sebanyak 50 kali.

1) RNN: Pada percobaan ini arsitektur *Recurrent Neural Network* yang akan dipakai adalah *SimpleRNN* dari *library Keras. Dropout* yang digunakan adalah *SpatialDropout1D* dimana berfungsi sebagai *Visible Layes* sebelum *Input Layer* dan diberi nilai 0.5. Pada layer *SimpleRNN* akan diberikan *input unit* yang berjumlah 16. Gambar 5 adalah arsitektur *RNN* yang digunakan.

2) LSTM: Pada percobaan kali ini arsitektur *Long Short-Term Memory* yang akan dipakai adalah *LSTM* dari *library Keras. Dropout* yang digunakan adalah *SpatialDropout1D* dimana berfungsi sebagai *Visible Layes* sebelum *Input Layer* dan diberi nilai 0.5. Pada arsitektur ini *input* dari *layer LSTM* diberikan nilai 128. Gambar 5 adalah arsitektur *LSTM* yang digunakan.

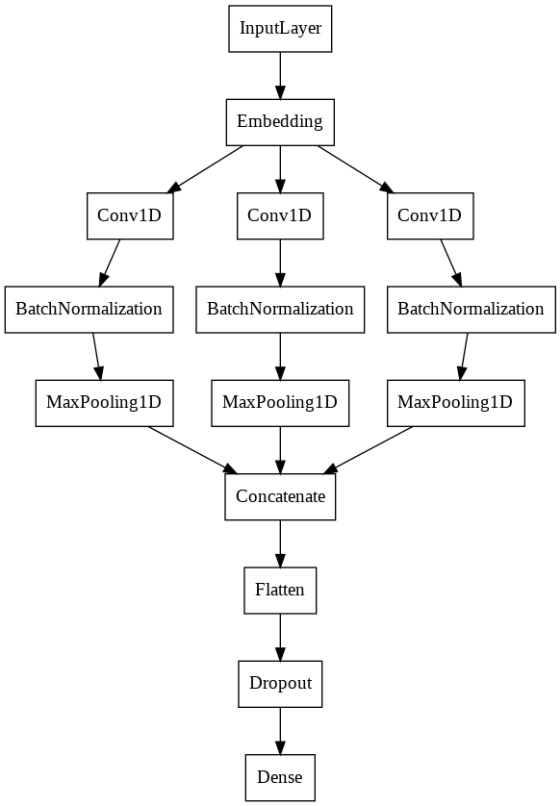
3) GRU: Pada percobaan kali ini arsitektur *Gated Recurrent Unit* yang akan dipakai adalah *GRU* dari *library Keras*. *Dropout* yang digunakan adalah *SpatialDropout1D* dimana berfungsi sebagai *Visible Layes* sebelum *Input Layer* dan diberi nilai 0.5. Pada arsitektur ini *input* dari *layer GRU* diberikan 16 jumlah unit. Gambar 5 adalah arsitektur *GRU* yang digunakan.



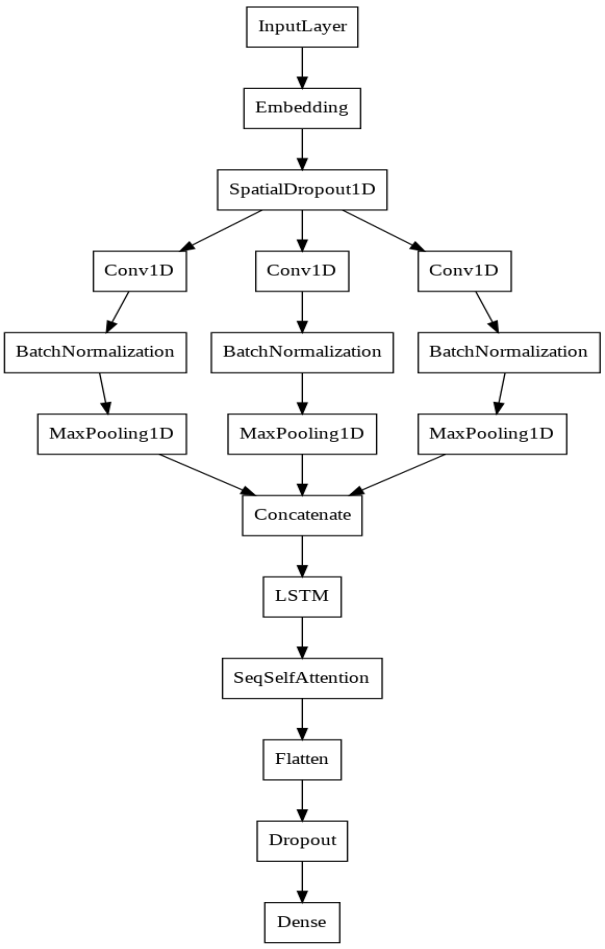
Gmbar 5. Arsitektur *RNN, LSTM,* dan *GRU*

4) CNN: Pada percobaan kali ini arsitektur *Convolutional Neural Network* yang akan dipakai adalah *CNN* dari *library Keras*. Pada arsitektur ini *Convolutional Layer* yang digunakan berjumlah 3-unit dengan *Batch Normalization* yang berfungsi untuk standarisasi *input* dan *Maxpooling* pada setiap unitnya yang kemudian setiap unit akan disatukan pada *Concatenate Layer* dan akan dijadikan *single array* pada *layer Flatten*. Gambar 6 adalah arsitektur *CNN* yang digunakan.

5) CNN-LSTM: Pada percobaan kali ini arsitektur *CNN-LSTM* yang akan dipakai adalah *CNN* dari *library Keras* dan akan ditambahkan *layer LSTM* dari *library keras.* Pada arsitektur ini *Convolutional layer* yang digunakan memakai *setting* yang sama dengan *setting* parameter dari arsitektur *CNN* yang dilakukan sebelumnya dengan tambahan dan akan disatukan pada *concatenate layer* kemudian akan dimasukkan sebagai input dari *LSTM layer*. Hanya saja sebelum *Convolutional Layer* akan diberi *SpatialDropout1D* sebagai *Visible Layer*. *Setting* parameter dari *LSTM layer* akan mengikuti *setting* dari arsitektur *LSTM* yang dilakukan sebelumnya yang akan dimasukkan pada *layer attention* dan dijadikan *single array* pada *layer Flatten*. Gambar 7 memperlihatkan arsitektur *CNN-LSTM* yang digunakan dalam penelitian ini.



Gambar 6. Arsitektur *CNN*



Gambar 7. Arsitektur *CNN-LSTM*

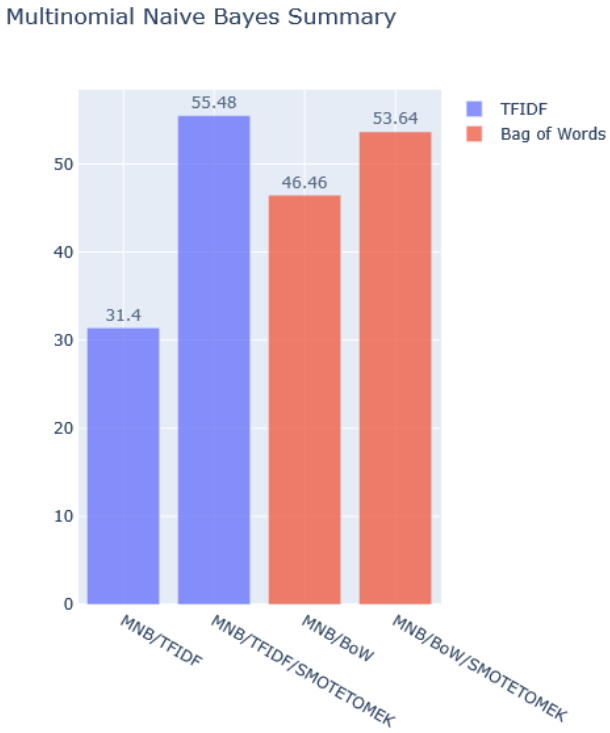
VI. Hasil Eksperimen dan Evaluasi

Bagian ini akan membahas hasil eksperimen yang dilakukan dan evaluasi dari setiap model yang digunakan.

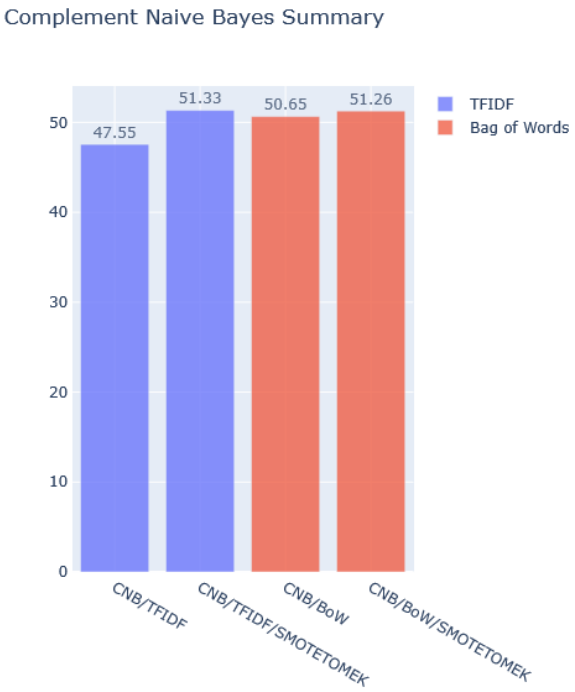
A. *Naïve Bayes*

Pada metode dengan algoritma klasifikasi *Multinomial Naïve Bayes* dengan ekstraksi fitur *Bag-of-Words* tanpa *resampling* data menghasilkan model dengan skor *F1* sebesar 46.46 dan dengan *resampling* metode *SMOTETomek* pada data menghasilkan model dengan skor *F1* sebesar 53.64. Sedangkan pada ekstraksi fitur *TF-IDF* dan algoritma klasifikasi *Multinomial Naïve Bayes* dengan data tanpa *resampling* menghasilkan model dengan skor *F1* sebesar 31.40 dan dengan data setelah pengaplikasian *resampling* menghasilkan model dengan skor *F1* sebesar 55.48 Gambar 8 adalah hasil perbandingan model dari algoritma klasifikasi *Multinomial Naïve Bayes*.

Pada metode *Complement Naïve Bayes* dengan esktraksi fitur *Bag-of-Words* tanpa *resampling* data mendapatkan 50.65 dan dengan *resampling* data menghasilkan model dengan skor *F1* sebesar 51.26. Sedangkan pada ekstraksi fitur *TF-IDF* pada algoritma klasifikasi *Complement Naïve Bayes* tanpa *resampling* data menghasilkan model dengan skor *F1* sebesar 47.55 dan dengan *resampling* data menghasilkan model dengan skor *F1* sebesar 51.33. Gambar 9 adalah hasil perbandingan model dari algoritma klasifikasi *Complement Naïve Bayes.*



Gambar 8. Perbandingan model *Multinomial Naïve Bayes MNB/TFIDF* dan *MNB/BoW* dengan *resampling* data *SMOTETOMEK*

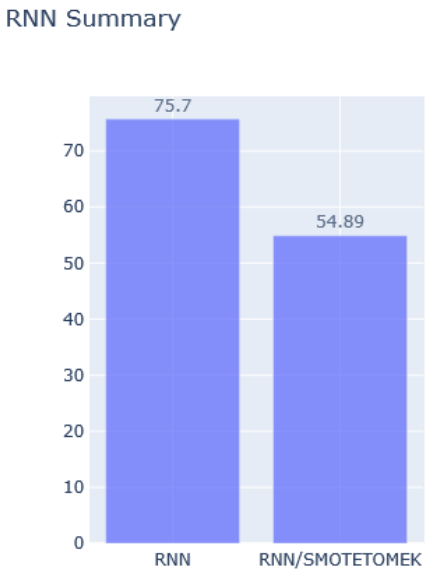


p-ISSN : 2443-2210
e-ISSN : 2443-2229

Gambar 9. Perbandingan model *Complement Naïve Bayes CNB/TFIDF* dan *CNB/BoW* dengan *resampling* data *SMOTETOMEK*

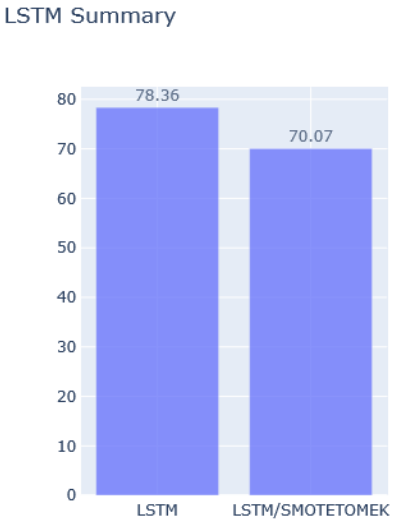
B. Neural Network

Pada metode *Neural Network* dengan arsitektur *Recurrent Neural Network* dengan *input* tanpa *resampling* data menghasilkan model dengan skor *F1* sebesar 75.70 dan dengan *input* setelah *resampling* data menggunakan metode *SMOTETomek* menghasilkan model dengan skor *F1* sebesar 54.89. Gambar 10 adalah hasil perbandingan model dari *Recurrent Neural Network*.



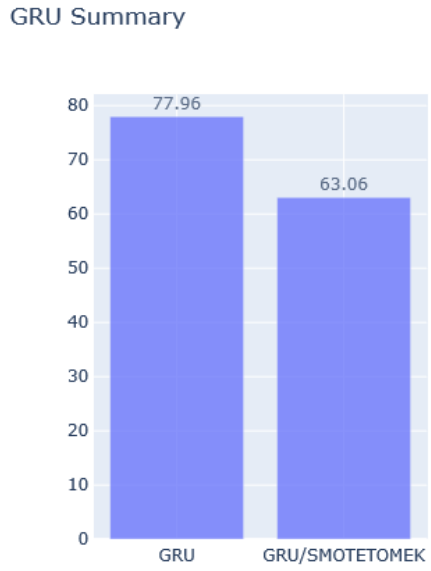
Gambar 10. Perbandingan model *RNN* dengan *resampling* data *SMOTETOMEK*

Pada arsitektur *Long Short-Term Memory* dengan *input* tanpa *resampling* data menghasilkan model dengan skor *F1* sebesar 78.36 dan dengan *input* setelah *resampling* data menghasilkan model dengan skor *F1* sebesar 70.07. Gambar 11 adalah hasil perbandingan model dari *Long Short-Term Memory*.



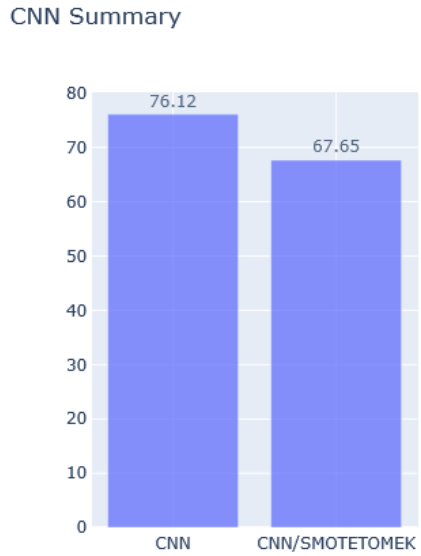
Gambar 11. Perbandingan model *LSTM* dengan *resampling* data *SMOTETOMEK*

Pada arsitektur *Gated Recurrent Unit* dengan *input* tanpa *resampling* data menghasilkan model dengan skor *F1* sebesar 77.96, dan dengan *input* setelah *resampling* data menghasilkan model dengan skor *F1* sebesar 63.06. Gambar 12 adalah hasil perbandingan model dari *Gated Recurrent Unit*.



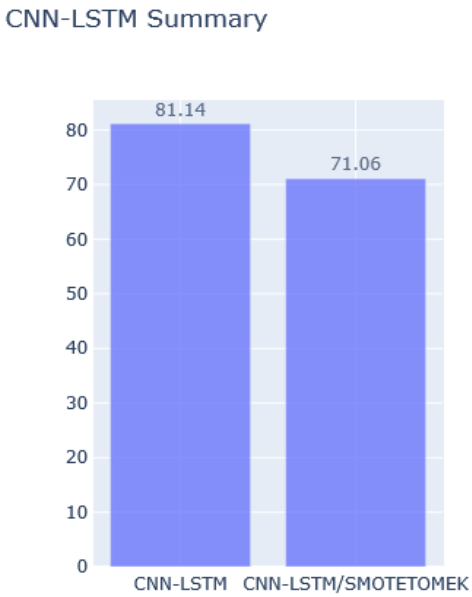
Gambar 12. Perbandingan model *GRU* dengan *resampling* data *SMOTETOMEK*

Pada arsitektur *Convolutional Neural Network* dengan *input* tanpa *resampling* data menghasilkan model dengan skor *F1* sebesar 76.12 dan dengan *input* setelah *resampling* data menghasilkan model dengan skor *F1* sebesar 67.65. Gambar 13 adalah hasil perbandingan model dari *Convolutional Neural Network*.



Gambar 13. Perbandingan model *CNN* dengan *resampling* data *SMOTETOMEK*

Pada arsitektur *Neural Network* gabungan *Convolutional Neural Network* dan *Long Short-Term Memory* dengan *input* tanpa *resampling* data menghasilkan model dengan skor *F1* sebesar 81.14 dan dengan *input* setelah *resampling* data menghasilkan model dengan skor *F1* sebesar 71.06. Gambar 14 adalah hasil perbandingan model dari *CNN-LSTM*.



Gambar 14. Perbandingan model *CNN-LSTM* dengan *resampling* data *SMOTETOMEK*

VII. DISKUSI HASIL

Tabel IV adalah tabel perbandingan hasil dari setiap model dari penelitian ini. Model dengan algoritma klasifikasi *Naïve Bayes* pada penelitian ini mendapatkan *F1* skor terbaik sebesar 55.48 pada model kombinasi dari *Multinomial Naïve Bayes* dengan fitur ekstraksi *TF-IDF* dan *resampling* data. Sedangkan pada model dengan arsitektur *Neural Network* pada penelitian ini mendapatkan skor *F1* terbaik sebesar 81.14 pada model dengan arsitektur *CNN-LSTM*.

Selain itu berdasarkan hasil penelitian pada Tabel IV model *Multinomial Naïve Bayes* dengan ekstraksi fitur *TF-IDF* adalah model dengan skor *F1* yang paling rendah. Hal ini disebabkan karena model gagal dalam memprediksi kelas minoritas. Hasil dari performa algoritma klasifikasi *Naïve Bayes* yang dilakukan menghasilkan model dengan skor *F1* yang kurang baik dikarenakan normalisasi dari *dataset* yang kurang seragam sehingga mempengaruhi hasil dari ekstraksi fitur yang berifat frekuensi dari tiap kata yang ada pada *dataset*, sementara pada arsitektur *Neural Network*, hasil cenderung lebih baik karena ekstraksi fitur pada *Word Embedding Vector Pre-trained* dari *FastText* mengekstraksi tiap kata dengan *n-gram* untuk mempelajari representasi dari setiap kata [37] yang ada pada *dataset* sehingga efek dari kurang seragamnya tiap kata dalam *dataset* dapat diminimalisir.

TABEL IV
HASIL PERBANDINGAN MODEL

Metode	F1 Score
MNB/TF-IDF	31.40
MNB/TF-IDF/SMOTETomek	**55.48**
MNB/Bag-of-Words	46.46
MNB/Bag-of-Words/SMOTETomek	53.64
CNB/TF-IDF	47.55
CNB/TF-IDF/SMOTETomek	51.33
CNB/Bag-of-Words	50.65
CNB/Bag-of-Words /SMOTETomek	51.26
RNN	75.70
RNN/SMOTETomek	54.89
LSTM	78.36
LSTM/SMOTETomek	70.07
GRU	77.96
GRU/SMOTETomek	63.06
CNN	76.12
CNN/SMOTETomek	67.65
CNN-LSTM	**81.14**
CNN-LSTM/SMOTETomek	71.06

Untuk menguji lebih lanjut terkait skor *F1* pada model dengan hasil terbaik pada algoritma klasifikasi *Naïve Bayes* dan arsitektur *Neural Network* akan dilakukan uji signifikansi. *Input* pengujian berulang akan berupa hasil skor *F1* dari 10 kali *StratifiedKFold Cross Validation* untuk *Multinomial Naïve Bayes* dan 10 *epoch* untuk *CNN-LSTM*, dengan nilai keyakinan yang ditetapkan sebesar 90%. Hasil dari uji signifikansi dari pengujian berulang dapat dilihat pada Tabel V. Dimana pada pengujian menghasilkan skor *student t-test* sebesar 8.412E-16. Nilai tersebut lebih kecil dibandingkan nilai keyakinan yang ditetapkan yaitu 90%. Dengan demikian terdapat perbedaan yang signifikan antara skor *F1* dari skor terbaik algoritma klasifikasi *Naïve Bayes* dengan skor terbaik dari arsitektur *Neural Network*.

TABEL V
SIGNIFIKAN TEST

	Multinomial Naïve Bayes	**CNN-LSTM**
Percobaan ke	**Skor F1 - MNB**	**Skor F1 - CNN-LSTM**
1	53,97	73,16
2	53,87	80,28
3	57,19	81,19
4	52,93	81,82
5	53,52	81,30
6	55,58	81,18
7	55,27	80,92
8	57,77	82,10
9	53,62	82,00
10	54,52	81,13
Rata-rata	54,82	80,51
t-test	8,41205E-16	
Alpha	0,1	

Hasil *resampling* dengan metode *SMOTETomek* pada algoritma klasifikasi *Naïve Bayes* memiliki pengaruh dalam menaikan nilai skor F1, terlebih lagi pada algoritma *Multinomial Naïve Bayes* dimana pertambahan dari persentasi skor *F1* lebih banyak dibandingan *Complement Naïve Bayes* dan juga *resampling* data menghasilkan performa yang lebih baik jika dikombinasikan dengan fitur ekstraksi *TF-IDF*. Selain itu, dengan kombinasi estraksi fitur *TF-IDF* dan *resampling* data *SMOTETomek*, algoritma klasifikasi *Multinomial Naïve Bayes* dapat menghasilkan model dengan skor *F1* yang lebih tinggi dibandingkan model dengan algoritma klasifikasi *Complement Naïve Bayes*, dimana algoritma klasifikasi *Complement Naïve Bayes* adalah algoritma klasifikasi yang cenderung lebih cocok dalam menangani data dengan kelas yang tidak seimbang [7]. Hasil dari resampling data pada *Complement Naïve Bayes* dengan ekstraksi fitur *Bag-of-Words* tidak menunjukan hasil dengan peningkatan skor *F1* seperti pada *Multinomial Naïve Bayes* terjadi karena cara kerja dari *Complement Naïve Bayes* sendiri dimana *Complement Naïve Bayes* menggunakan data *train* dari setiap kelasnya dengan seimbang [5] sehingga penyeimbangan *dataset* sudah dilakukan terlebih dahulu sebelum proses *resampling* data terjadi.

Sedangkan pada hasil penelitian dengan menggunakan arsitektur *Neural Network*, hasil dari *input* dengan *resampling* data menghasilkan model dengan skor *F1* yang lebih kecil dibandingkan tanpa *resampling* data. Hal ini berbanding terbalik jika dibandingkan dengan hasil dari algoritma klasifikasi *Naïve Bayes*. Di samping itu, berdasarkan hasil penelitian yang didapat, pengaplikasian metode *resampling SMOTETomek* pada arsitektur *Neural Network* paling berpengaruh pada arsitektur *CNN* dan *LSTM* dimana kedua arsitektur tersebut memiliki selisih skor *F1* dengan *input* tanpa *resampling* dan *input* setelah *resampling* yang lebih sedikit dibandingkan arsitektur lainnya.

VIII. KESIMPULAN

Pembuatan model analisis sentimen berbahasa Indonesia dengan multi *keyword* sebagai topik dengan metode algoritma klasifikasi *Multinomial Naïve Bayes* dan *Complement Naïve Bayes* sudah berhasil dibuat. Selain itu sebagai pembanding model analisis sentimen dengan arsitektur *Neural Network* yaitu *Recurrent Neural Network, Long Short-Term Memory, Gated Recurrent Unit, Convolutional Neural Network*, serta kombinasi dari *Convolutional Neural Network* dan *Long Short-Term Memory* pun sudah berhasil dibuat. Hasil yang didapat adalah hampir semua model analisis sentimen dengan arsitektur *Neural Network* memiliki skor yang lebih baik dibandingkan model dengan algoritma klasifikasi *Naïve Bayes*. Hasil yang berbeda didapatkan dari model dengan arsitektur *Recurrent Neural Network* dengan *input* data setelah resampling menggunakan metode *SMOTETomek* dimana hasil skornya lebih kecil dari *Multinomial Naïve Bayes* dengan ekstraksi fitur *TF-IDF* dan *resampling* data menggunakan metode *SMOTETomek*.

Selain itu, hasil *resampling* data pun sudah dapat diaplikasikan serta diamati perbedaan tiap efeknya dengan algoritma klasifikasi *Naïve Bayes* maupun arsitektur *Neural Network*. Dimana efek dari *resampling* data terhadap skor *F1* lebih berpengaruh terhadap algoritma klasifikasi *Naïve Bayes* dibandingkan arsitektur *Neural Network*.

IX. SARAN

Pada penelitian yang lebih lanjut akan dapat menggunakan *dataset* berbahasa Indonesia dengan data yang lebih banyak dengan label sentimen yang lebih sesuai dengan ilmu linguistic agar dapat meminimalisir bias dari *manual labeling*. Selain itu data perkelas yang ada pada *dataset* juga dapat diperbanyak dengan lebih banyaknya jumlah data *Tweets* pada *dataset* sehingga dapat menambah jumlah kosakata pada kelas minoritas. Pada *setting* parameter dapat dicoba untuk mencari parameter yang paling optimal berdasarkan parameter yang ada pada algoritma klasifikasi *Naïve Bayes* dan fitur ekstraksinya, juga parameter yang ada pada arsitektur *Neural Network* untuk mencapai akurasi yang lebih optimal. Selain itu dapat dicoba untuk melakukan analisis sentimen yang dapat mendeteksi sarkasme, *emoticon*, dan negasi. Sedangkan pada *Neural Network* dapat dicoba untuk melakukan klasifikasi dengan ekstraksi fitur lainnya.

DAFTAR PUSTAKA

[1] L. J. Sheela, "A Review of Sentiment Analysis in Twitter Data Using Hadoop," *International Journal of Database Theory and Application*, vol. 9(1), pp. 77–86, May. 2016.
[2] B. Liu, *Sentiment Analysis and Subjectivity Handbook of Natural Language Processing*, Cambridge, United Kingdom: Cambridge University Press, 2015.
[3] P. N. Tan, M. Steinbach, & V. Kumar, *Introduction to Data Mining*. Boston, United States: Pearson Education, 2006.

[4] I. Rish, "An empirical study of The Naive Bayes Classifier," *IJCAI 2001 workshop on empirical methods in artificial intelligence,* vol. 3, pp. 41–46, Aug. 2001.
[5] Jason. D. M. Rennie, S. Lawrence, T. Jaime, & R. K. David, "Tackling the Poor Assumptions of Naive Bayes Text Classifiers," *ICML'03 Proceedings of the Twentieth International Conference on Machine Learning*, pp. 616–623, Aug. 2003.
[6] I. H. Witten, F. Eibe & M. A. Hall, *Data mining: Practical Machine Learning Tools and Techniques*. Third Edition. USA: Elsevier, 2011.
[7] (2020) Scikit-learn website. [Online]. Tersedia: https://scikit-learn.org/stable/modules/generated/sklearn.naive_bayes.ComplementNB.html/
[8] P. Munish, S. Aum, P. Timothy, Tipparaju M. Srivinas, Moreno W. Alejandro, & P. Yashwant, "Introduction to Artificial Neural Network (ANN) as a Predictive Tool for Drug Design, Discovery, Delivery, and Disposition: Basic Concepts and Modeling," *Artificial Neural Network for Drug Design, Delivery and Disposition*, pp. 3–13, Oct. 2015.
[9] R. Priyadarshini, N. Dash, T. Swarnkar, & R. Misra, "Functional Analysis of Artificial Neural Network for *Dataset* Classification," *ACCTA-2010 Special Issue of IJCCT Vol.1 Issue 2, 3, 4; 2010 for International Conference*, Aug. 2010
[10] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, & G. Jianfeng, "Deep Learning Based Text Classification: A Comprehensive Review," 2020; arXiv:2004.03705.
[11] Tatyana I. Poznyak, Isaac Chairez Oria, & Alexander S. *Poznyak, Ozonation and Biodegradation in Environmental Engineering: Dynamic Neural Network Approach*. Elsevier, 2018.
[12] R. DiPietro, & G. D. Hager, *Handbook of Medical Image Computing and Computer Assisted Intervention*, Chapter 21. pp. 503–519. Academic Press. 2019.
[13] R. Pascanu, T. Mikolov, & Y. Bengio, "On the difficulty of training recurrent neural networks," *ICML'13: Proceedings of the 30th International Conference on International Conference on Machine Learning*, pp. III-1310–III-1318, June. 2013.
[14] S. Hochreiter, J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol 9(8), pp. 1735-1780, Nov. 1997.
[15] K. Cho, B. van Merrienboer, D. Bahdanau, & Y. Bengio, "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches," 2014; arXiv:1409.1259.
[16] J. Chung, C. Gulcehre, K. Cho, & Y. Bengio, "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," 2014; arXiv:1412.3555.
[17] Y. Lecun, "Generalization and network design strategies," *Connectionism in perspective*, pp. 143–155, June. 1989.
[18] A. Krizhevsky, I. Sutskever, & G. E. Hinton, *Imagenet classification with deep convolutional neural networks*, F. Pereira, C. J. C. Burges, L. Bottou, & K. Q. Weinberger. Advances in Neural Information Processing Systems 25. pp. 1097–1105, Curran Associates, Inc, 2012.
[19] Y. Zhou, & R. Chellappa, "Computation of optical flow using a neural network," *IEEE 1988 International Conference on Neural Networks,"* pp. 71–78, July. 1988.
[20] A. Ghods, & D. J. Cook, "A Survey of Techniques All Classifiers Can Learn from Deep Networks: Models, Optimizations, and Regularization," 2019; arXiv:1909.04791.
[21] C. Zhou, C. Sun, Z. Liu, & Francis C. M. Lau, "A C-LSTM Neural Network for Text Classification," 2015; arXiv: 1511.08630.
[22] H. He, & Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications. Wiley-IEEE Press, 2013.
[23] (2020) GetOldTweets3 pip website [Online]. Tersedia; https://pypi.org/project/GetOldTweets3/
[24] (2020) Twitter Developer website [Online]. Tersedia; https://developer.twitter.com/en/docs/basics/rate-limiting/
[25] Asniar, & B R Aditya. "A Framework for Sentiment Analysis Implementation of Indonesian Language Tweet on Twitter," *Journal of Physics: Conference Series, vol. 801(1), article id. 012065*. Jan. 2017.
[26] R. T. W. Lo, B. He, & I. Ounis. "Automatically building a stopword list for an information retrieval system," *J. Digit. Inf. Manaag*, vol. 5, pp. 17–24. Jan. 2005.
[27] H. Saif, M. Fernandez, Y. He, & H. Alani. "On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter," *LREC'14 Proceedings of the Ninth International Conference on Language Resources and Evaluation*, pp. 810–817. May. 2014
[28] C. T. Meadow, B. R. Boyce, & D. H. Kraft, *Text information retrieval systems*, 2nd ed. San Diego, California: Academic Press, 2000.
[29] W. Frakes, *Information Retrieval: Data Structures and Algorithms*, Englewood Cliffs, New Jersey: Prentice-Hall, 1992.
[30] Z. A. Bakar, & N. A. Rahman. *Evaluating the effectiveness of thesaurus and stemming methods in retrieving Malay translated Al-Quran documents*, seri Lecture Notes in Computer Science. Digital Libraries: Technology and Management of Indigenous Knowledge for Global Access. pp. 653–662. Springer. 2003.
[31] C. Orˇasan, V. Pekar, & L. Hasler. "A comparison of summarisation methods based onterm specificity estimation," *LREC'04 Proceedings of the Fourth International Conference on Language Resources and Evaluation*. pp. 1037–1041. May. 2004.
[32] T. Gaustad, & G. Bouma. *Computational Linguistics in the Netherlands:* Accurate stemming of Dutch for text classification, seri Language and Computers: Studies in Practical Linguistics. Amsterdam: Rodopi, 2002.
[33] (2020) PySastrawi Github website [Online] Tersedia: https://github.com/har07/PySastrawi/
[34] (2020) Sastrawi Github website [Online] Tersedia: https://github.com/sastrawi/sastrawi/
[35] R. N. Waykole, & A. D. Thakare. "a Review of Feature Extraction Methods for Text Classification," *International Journal of Advance Engineering and Research Development*, vol.5(4), pp. 351–354, Apr. 2018.
[36] (2020) FastText website [Online] Tersedia: https://fasttext.cc/docs/en/faqs.html/
[37] P. Bojanowski, E. Grave, A. Joulin, & T. Mikolov, "Enriching Word Vectors with Subword Information," 2016; arXiv:1607.04606.
[38] S. Raschka, "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning," 2018; arXiv:1811.12808.
[39] M. S. Santos, J. P. Soares, P. H. Abreu, H. Araujo, & J. Santos, "Cross-Validation for Imbalanced *Datasets*: Avoiding Overoptimistic and Overfitting Approaches," *IEEE Computational Intelligence Magazine*, vol.13(4), pp.59–76. Nov. 2018.
[40] N. V. Chawla, K. W. Bowyer, L. O. Hall, & W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol.16, pp.321–357. Jun. 2002.
[41] M. Bekkar, & T. A. Alitouche, "Imbalanced Data Learning Approaches Review," *IJDKP International Journal of Data Mining & Knowledge Management Process*, vol.3(4), pp.15–33, Jul. 2013.