Klasifikasi Topik Pada Tweet Berbahasa Indonesia menggunakan Ekspansi fitur Fast-Text dengan Metode *Support Vector Machine* (SVM)

1st Imaduddin Muhammad Fadhil
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
imfadhil@students.telkomuniversity.ac.id

2nd Yuliant Sibaroni S
Fakultas Informatika
Universitas Telkom
Bandung, Indonesia
yuliant@telkomuniversity.ac.id

Abstrak
Twitter merupakan platform sosial media populer yang memberi pengguna kemampuan untuk mengirim pesan teks dengan panjang maksimum 280 karakter yang menyebabkan banyak penggunaan variasi kata yang menyebabkan kesalahan penulisan kosakata dan saat ini semakin banyak *tweet* yang tersebar dan karena penyebaran yang sangat pesat menyebabkan kelebihan informasi. Dari permasalahan yang diangkat diperlukan kemampuan untuk mengenali kata yang memiliki kesalahan dalam penulisan dan mengkategorikan *tweet* kedalam kategori tertentu. Oleh karena itu, penelitian ini bertujuan untuk membangun sistem klasifikasi topik pada tweet yang dapat mempelajari kesalahan penulisan pada suatu kata dan Ekspansi fitur menggunakan vektor kata *pretrained* dari FastText dapat digunakan untuk mengenali kesalahan penulisan pada suatu kata karena proses pembangunan vektor kata dari FastText yang dapat mempelajari struktur internal dari sebuah kata yang akan dimanfaatkan pada model klasifikasi *Support Vector Machine*. Hasil terbaik dari penelitian ini mendapatkan akurasi sebesar 76.88% dengan penerapan ekspansi fitur pada top-1 namun penerapan ekspansi fitur menggunakan vektor kata *pretrained* Bahasa Indonesia pada top-2 sampai top-10 menurunkan kinerja klasifikasi *Support Vector Machine*.

Kata kunci : klasifikasi topik, ekspansi fitur, FastText, *Support Vector Machine*

Abstract
Twitter is a popular social media platform that gives users the ability to send text messages with a maximum length of 280 characters which causes a lot of use of word variations that cause vocabulary writing errors and nowadays more and more tweets are spread and because of the very rapid spread it causes information overload. From the problems raised, it requires the ability to recognize words that have errors in writing and categorize tweets into certain categories. Therefore, this study aims to build a topic classification system on tweets that can study writing errors in a word and feature expansion using pretrained word vectors from FastText can be used to recognize writing errors in a word because the process of building word vectors from FastText can study the structure internal of a word that will be used in the Support Vector Machine classification model. The best results from this study get an accuracy of 76.88% with the application of feature expansion on top-1 but application of feature expansion using Indonesian pretrained word vectors on top-2 to top-10 reduces the performance of the Support Vector Machine classification.

Keywords: Topic Classification, Feature Expansion, FastText, Support Vector Machine

I. PENDAHULUAN

Twitter merupakan layanan mikro-blog yang memberikan pengguna kemampuan untuk mengirim pesan teks dengan panjang maksimum 280 karakter [1]. Saat ini, semakin banyak *tweet* atau informasi yang menyebar di internet. Namun dengan meluasnya penyebaran informasi tersebut menyebabkan adanya kelebihan informasi. Kemampuan untuk mengklasifikasikan *tweet* ke dalam kategori tertentu akan membantu untuk menangani informasi yang berlebihan di *Twitter*. Hal tersebut sangat menarik untuk penelitian di bidang klasifikasi teks [2].

Penelitian ini tidak lepas dari penelitian-penelitian terdahulu untuk bahan perbandungan dan bahan kajian. Penelitian terkait ekspansi fitur dan FastText *pretrained* vektor kata sebelumnya sudah pernah dilakukan seperti pada penelitian [3] percobaan terkait ekspansi fitur dilakukan menggunakan *Suport Vector Machine*, Naïve Bayes, dan *Logistic Regression* dengan mengaplikasikan ekspansi fitur menggunakan *Word2Vec*. Pada penelitian [3] didapatkan kesimpulan bahwa

algoritma *Support Vector Machine* dengan mempunyai kinerja baik namun menurun apabila menerapkan perluasan fitur menggunakan *Word2Vec* [3] dan didalam penelitian [4] diketahui bahwa vektor kata *pretrained* dari FastText dapat digunakan untuk mengenali kata yang mempunyai penulisan yang salah. Berdasarkan penelitian sebelumnya diperlukan riset seperti yang diusulkan dalam penelitian ini dengan menggunakan metode fitur ekspansi FastText pada algoritma *Support Vector Machine* (SVM).

Seperti yang telah diketahui didalam penelitian [3][4]. Pada penelitian sebelumnya belum ada penelitian terkait implementasi ekspansi fitur menggunakan vektor kata *pretrained* dari FastText pada metode klasifikasi *Support Vector Machine*, hal tersebut membedakan penelitian ini dengan penelitian sebelumnya.

Tujuan dari penelitian ini adalah untuk mengetahui skenario split data, kombinasi n-gram dan jumlah maksimal fitur terbaik, lalu mengetahui parameter dan kernel terbaik dari model *Support Vector Machine* (SVM) dan performa dari metode pengklasifikasian *Support Vector Machine* (SVM) dengan implementasi ekspansi fitur menggunakan vektor kata *pretrained* dari FastText. Penelitian ini menggunakan FastText karena FastText mempunyai performa yang baik, memungkinkan sufiks dan prefiks kata juga dapat melatih model untuk data yang besar dengan cepat [5]. Batasan dalam penelitian ini yaitu *tweet* yang digunakan adalah *tweet* berbahasa Indonesia dan metode ekspansi fitur menggunakan model *pre-trained* berbahasa Indonesia dari FastText dan pengklasifikasiian menggunakan algoritma *Support Vector Machine* (SVM).

II. KAJIAN TEORI

Topik dari penelitian ini mengacu kepada penelitian [3] yang bertujuan untuk mengurangi ketidakcocokan kosakata pada klasifikasi topik pada *tweet* dengan menggunakan ekspansi fitur Word2Vec. Penelitian sebelumnya menetapkan 19 topik yang mempunyai jumlah *keyword* yang bermacam-macam. Dalam hal metode, penelitian sebelumnya menggunakan berbagai teknik pembelajaran terkenal untuk mengklasifikasikan topik. Teknik ini meliputi: Naïve Bayes, *Support Vector Machine* (SVM), dan *Logistic Regression*[3]. Hasil dari penelitian [3] menunjukkan bahwa algoritma *Logistic Regression* Memiliki hasil akurasi tertinggi 58.86%, diikuti oleh *Support Vector Machine* (SVM) 54.67%, dan Naïve Baiyes 52.99% jika menggunakan korpus dari *Google News* dari hasil tersebut terbukti bahwa algoritma *Support Vector Machine* (SVM) memiliki kinerja yang menurun karena menerapkan ekspansi fitur memakai *word embedding* dari Word2Vec dibandingkan penemuan sebelumnya yang selalu memiliki kinerja yang terbaik [3].

Selain penelitian [3] topik tentang ekspansi fitur pernah dilakukan juga sebelumnya pada penelitian [6] dengan menggunakan kombinasi dari Bags-of-Words (BOW) dengan pendekatan *Lexical Onthology* yang diklasifikasikan menggunakan model *Naïve Bayes*, *Support Vector Machine* (SVM), dan *Decision Tree*. Dari penelitian [6] diketahui bahwa *Support Vector Machine* (SVM) mempunyai akurasi yang paling baik sebesar 61.59% dibandingkan dengan Naïve Bayes (46.77%) dan Decision Tree (37.26%).

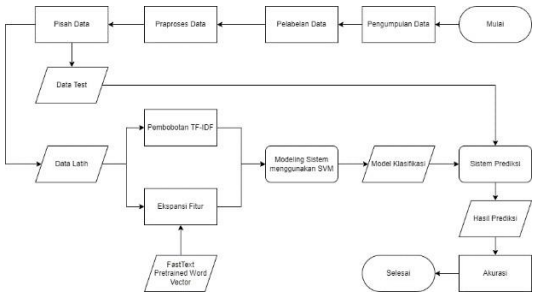
Penelitian ini mempunyai fokus untuk mendapatkan performansi *Support Vector Machine* (SVM) terhadap ekspansi fitur penggunaan FastText digunakan karena pada penelitian [5] FastText dapat digunakan untuk mencari kata yang berhubungan secara semantik dari kumpulan vektor kata untuk mengatasi kata yang mempunyai kesalahan dalam penulisan dengan mengetahui *cosine similarity* dari sebuah kata. Penggunaan FastText diperlukan vektor kata dan dalam penelitian [7] melakukan pelatihan vektor kata terhadap 157 bahasa termasuk Bahasa Indonesia menggunakan data dari Wikipedia dan Common Crawl.

Pembahasan terkait vektor kata *pretrained* dari FastText untuk Bahasa Indonesia sebelumnya sudah pernah dibahas pada penelitian [4] dimana tujuan dari penelitian ini untuk mendalami vektor kata *pretrained* dari FastText yang telah dibangun pada penelitian [7] dengan kesimpulan bahwa vektor kata Bahasa Indonesia *pretrained* dari FastText tidak memuat semua kata yang ada didalam Kamus Besar Bahasa Indonesia (hanya terdapat sekitar 75%) namun penggunaan vektor kata *pretrained* masih dapat diandalkan dalam pemrosesan bahasa natural.

Berdasarkan penelitian yang telah dikembangkan sebelumnya [3][6], penulis akan mengimplementasikan metode klasifikasi *Support Vector Machine* (SVM) dengan akurasi sebagai parameter kinerja terhadap data yang telah dilakukan ekspansi fitur menggunakan vektor kata *pretrained* Bahasa Indonesia dari FastText yang diharapkan dapat mendapatkan akurasi yang lebih baik dalam kasus klasifikasi topik di *Twitter*[3].

III. METODE

Flowchart berikut dibuat untuk menjelaskan bagaimana gambaran umum dari sistem yang telah dibangun pada penelitian ini.



GAMBAR 1. Flochart Sistem Analisis

A. Pengambilan Data

Pengumpulan data *tweet* untuk keperluan penelitian ini dikumpulkan dengan Teknik *Crawling Twitter* Data menggunakan Bahasa Python melalui Twitter API (Application Program Interface) tweet yang diambil adalah *tweet* bahasa Indonesia yang berada pada periode September 2021 sampai Oktober 2021 dan mendapatkan 5356 *tweet* bahasa indonesia. Tabel 1 menjelaskan sampel *tweet* yang diambil.

TABEL 1. Sampel *Tweet*

Tweet
Sayangnya, besarnya biaya politik ini tak sebanding dengan hasil yang diraih, yakni terpilihnya pemimpin yang berkualitas.
@DG_Migueroz @KompasBola @peluitpanjangid @BaliUtd @beINSPORTSid Buat seluruh aktivitas Olahraga biar makin keren min. Apalagi desainnya yg unik pasti bikin eye catching.
Selamat pagi tweeps, awali pagi yang cerah ini dengan berdoa. Selamat beraktivitas, tetap semangat, dan selalu patuhi protokol kesehatan. Salam #IndonesiaRaya

B. Pelabelan Data

Tahap pelabelan data dilakukan oleh 2 (dua) orang untuk setiap *tweet* pelabelan dilakukan dengan cara membaca dan mempelajari setiap artikel dan mengumpulkan beberapa kata yang dapat menjadi petunjuk untuk melabelkan *tweet* dengan aturan untuk setiap *tweet* hanya memiliki 1 label dari 11 label yang telah ditentukan sebelumnya. Label dan distribusi topik dapat dilihat pada Tabel 2.

TABEL 2. Distribusi Topik

No	Label	Kuantitas	Presentase
1	**Olahraga**	634	11,8%
2	**Hiburan**	595	11,1%
3	**Pendidikan**	563	10,5%
4	**Teknologi**	549	10,3%
5	**Agama**	540	10,1%
6	**Budata**	525	9,8%
7	**Kesehatan**	517	9,7%
8	**Ekonomi**	508	9,5%
9	**Politik**	499	9,3%
10	**Hukum**	426	8,0%
11	**Motivasi**	394	7,4%
	Total	5751	

C. Praproses Data

Hasil data dari proses pelabelan diolah kembali dengan tujuan untuk mengubah data tekstual yang belum terstruktur menjadi terstruktur. Berikut merupakan tahapan praproses data yang akan dilakukan dalam penelitian ini, yaitu :

a. *Case Folding* : mengubah data tekstual yang mengandung huruf kapital menjadi huruf kecil.
b. *Normalization* : menghilangkan karakter khusus seperti *hashtag*, tanda baca, *mention*, *emoji*, *smiley*, angka dan juga mengubah kata "gaul" Bahasa Indonesia menjadi ejaan yang disempurnakan (EYD). Pada penelitian ini kata "gaul" diubah menggunakan kamus dari penelitian [8] yang terdiri dari 1309 kata.
c. Tokenisasi : mengubah data bahasa alami menjadi token yang lebih terstruktur dengan memecah data tekstual menjadi fragmen yang berisi data granular yang dipisahkan oleh spasi.
d. *Filtering* : penghapusan *stopwords* atau kata yang tidak mempunyai pengaruh dalam mengklasifikasi sebuah topik.
e. *Stemming* : *stemming* bertujuan untuk mengubah kata menjadi kata dasar dan *stemmer* yang digunakan pada penelitian ini merujuk pada penelitian [9].

D. Pembobotan TF-IDF

Data yang telah diproses sebelumnya akan dilakukan pembobotan untuk menghitung *rating* dari kata-kata yang ada. Metode tersebut akan menghitung bobot dari setiap kata t pada dokumen d dengan persamaan 1:

$$tfidf_t = f_{t,d} \times \log\frac{N}{df_t} \quad (1)$$

Dimana $tfidft$ adalah Bobot kata ke-t, ft,d adalah kemunculan kata t pada dokumen d, N adalah Total dokumen dan dft adalah Jumlah dokumen yang mengandung kata t[10]. Tabel 3 dan 4 menjelaskan ilustrasi perhitungan tf-idf untuk kata kunci "menang liga bola" dari 3 dokumen yang berbeda.

TABEL 3. Ilustrasi dokumen

Dokumen	Kalimat

ke-	
1	Persib menang liga sepak bola Indonesia
2	Teknik bola kunci tanding
3	Pemain bawa tim bola menang liga

TABEL 4. Ilustrasi perhitungan TF-IDF

Token	Kata Kunci	TF			df	N/df	IDF	Bobot		
		D1	D2	D3			$\log(\frac{N}{df})$	D1	D2	D3
Persib	0	1	0	0	1	3	0,477	0,477	0	0
Menang	1	1	0	1	2	1,5	0,176	0,176	0	0,176
Liga	1	1	0	1	2	1,5	0,176	0,176	0	0,176
Sepak	0	1	0	0	1	3	0,477	0,477	0	0
Bola	1	1	1	1	3	1	0	0	0	0
Indonesisa	0	1	0	0	1	3	0,477	0,477	0	0
Teknik	0	0	1	0	1	3	0,477	0	0,477	0
Bawa	0	0	0	1	1	3	0,477	0	0	0,477
Kunci	0	0	1	0	1	3	0,477	0	0,477	0
tanding	0	0	1	0	1	3	0,477	0	0,477	0
Pemain	0	0	0	1	1	3	0,477	0	0	0,477
Tim	0	0	0	1	1	3	0,477	0	0	0,477

E. Ekspansi Fitur FastText

FastText adalah metode *embedding* kata dan merupakan pengembangan dari metode Word2Vec, dimana n-gram dari sebuah karakter juga digunakan pada representasi kata. Representasi vektor dari sebuah kata diperoleh dengan mengambil jumlah vektor dari karakter n-gram yang muncul didalam kata tersebut namun kata yang utuh masih disertakan sebagai bagian dari katarker n-gram, sehingga model masih mempelajari satu vektor untuk setiap kata [5][11]. Penggunaan dari metode *embedding* adalah untuk mengelompokkan vektor kumpulan kata yang memiliki kesamaan dalam ruang vektor[3].

Vektor kata yang digunakan pada penelitian ini menggunakan vektor kata *pretrained* berbahasa Indonesia yang dapat digunakan dari penelitian [7] dimana korpus yang digunakan untuk melatih model tersebut berasal dari *Wikipedia* dan *Common Crawl* dengan jumlah kata yang terdapat pada vektor kata yang digunakan berjumlah 2 juta kata namun korpus yang digunakan belum menerapkan praproses data sehingga masih terdapat huruf kapital, tanda baca dan imbuhan didalam vektor kata[4].

Ide dari fitur ekspansi adalah untuk mengidentifikasi kata tidak berbobot pada representasi *tweet* dan mengganti nilai representasi kata tersebut dengan kata yang berhubungan secara semantik yang terdapat didalam vektor kata.

FastText mencari kata yang berhubungan secara semantik dengan menghitung nilai *cosine similarity* dari setiap pasangan n-gram yang berada didalam vektor kata dengan persamaan 2.

$$S(T,Q) = \frac{\sum_{i=1}^{n}(T_i \, x \, Q_i)}{\sqrt{\sum_{i=1}^{n}(T^2_i) \, x \, (Q^2_i)}} \quad (2)$$

Dimana T_i adalah nilai TF-IDF dari kata yang ada didalam tweet dan Q_i adalah nilai TF-IDF dari kata yang memiliki hubungan secara semantik, tabel 5 memperlihatkan kata yang berhubungan semantik dengan kata “Persib” dengan nilai n dari top-n adalah 10 rank-1 sampai rank-10 diurutkan berdasarkan besarnya *cosine similarity*

TABEL 5. Kata Serupa dari perdata.

Pada penelitian ini model *pre-trained* FastText digunakan untuk mengganti nilai dari kata yang tidak berbobot dengan nilai dari kata yang berhubungan secara semantik atau dapat disebut dengan ekspansi fitur, percobaan ekspansi fitur dilakukan berulang kali untuk mencari nilai n dari *top-n* terbaik yang mendefinisikan jumlah pengambilan kata yang berhubungan semantik. Algoritma ekspansi fitur yang digunakan untuk mengimplementasikan ekspansi fitur ditunjukkan pada algoritma 1.

Algoritma 1 Ekspansi Fitur (*dfVector*, *feature*, *ft*, *topn*) → *dataframe* teks vektor
//Mengaplikasikan ekspansi fitur untuk setiap fitur pada *dataframe* matriks vektor

Input **:** *dfVector* adalah dataframe dari matriks vektor data *tweet. feature* adalah fitur yang digunakan pada matriks vektor. *ft* adalah model vektor *pretrained* dari FastText. *Top-n* adalah jumlah kata bermakna serupa yang diambil.

a. **for** *each col* ***ϵ*** *dfVector.columns* **do**
b. *sim_word* ← *ft.similar_by_word(col, topn)*
c. **for** *each term* ***ϵ*** *sim_word* **do**
d. **if** *term* **=** *feature* ***then***
e. *dfVector[col] [(dfVector[col]=0)&(dfVector[term]≠0)]* ← *dfVector[term] [(dfVector[col]=0)&(dfVector[term]≠0)]*
f. **end if**
g. **end for**
h. **end for**

F. Klasifikasi *Support Vector Machine* (SVM)

Model *Support Vector Machine* (SVM) merepresentasikan data sebagai titik lalu memetakan titik tersebut kedalam n-dimensi ruang fitur yang terbagi kedalam beberapa kelas yang dipisahkan oleh hyperplane[10][13]. Sebuah *hyperplane* terbentuk dalam ruang fitur dengan memaksimalkan margin antara *hyperplane* dengan titik-titik yang terletak paling dekat dengannya sabagai vektor pendukung[14]. Untuk mendapatkan *hyperplane* yang dapat memisahkan data antar kelas pada penelitian ini menggunakan persamaan (3) [15].

$$\sum_{i g SV} y_i a_i K(x_i, x) + b \quad (3)$$

Dimana SV merupakan *support vector*, y_i merupakan label aktual, a_i merupakan koefisien ganda yang dibatasi oleh C (penalti), $K(x_i, x)$ adalah fungsi kernel dan b adalah parameter bebas. Pada penelitian ini dilakukan pencarian parameter C, gamma (didalam fungsi kernel) dan fungsi kernel terbaik untuk model klasfikasi yang dibuat dimana parameter C berfungsi untuk menentukan regularisasi atau margin, gamma adalah parameter yang mengatur jarak pengaruh dari suatu titik dan fungsi kernel berfungsi untuk mengembangkan *Support Vector Machine* (SVM) agar dapat memecahkan permasalahan non linear dengan melakukan pemetaan titik data kedalam dimensi yang lebih tinggi (dimensi fitur) dengan melakukan transformasi non-linier[15].

Kata	Rank-1	Rank-2	Rank-3	Rank-4	Rank-5
perdata	Perdata.	keperdataan	Perdata	hukum	perdatanya
	Rank-6	Rank-7	Rank-8	Rank-9	Rank-10
	KUHperdata	pidana	hukum. hukum	KHUPerdata	Perdata Hukum

G. Akurasi

Untuk mengevaluasi performa dari sistem yang telah selesai dilakukan dari tahapan sebelumnya akan di ukur akurasinya dan untuk validasi performa dalam penelitian ini menggunakan K-*Cross Validation* dengan K bernilai 5. Akurasi didapat dari model *confusion matrix* yang dapat dilihat pada tabel 6.

TABEL 6. *Confusion Matrix*

		Kelas Sebenarnya	
		Positive	Negative
Kelas Prediksi	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Confusion matrix digunakan untuk menghitung akurasi. Akurasi adalah nilai validitas model yang didapat dari jumlah klasifikasi benar dibagi dengan total klasifikasi yang dapat dilihat dari persamaan (4):

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

IV. HASIL DAN PEMBAHASAN

A. Hasil Pengujian

Tujuan dari penelitian ini adalah untuk mengetahui performansi dari metode klasifikasi *Support Vector Machine* (SVM) terhadap ekspansi fitur menggunakan vektor kata *pretrained* dari FastText. Performansi dihitung dari akurasi dalam klasifikasi yang didefinisikan sebagai presentase

keberhasilan pengklasifikasian *tweet* terhadap topik dan divalidasi menggunakan 5-*fold cross validation*. Untuk mendapatkan tujuan dari penelitian, tabel 7 menjelaskan 3 (tiga) skenario percobaan yang dilakukan pada penelitian ini.

TABEL 7. Skenario Pengujian

Skenario ke-	Perlakuan	Tujuan
1	Pengujian Model klasifikasi terhadap skenario pisah data, jumlah n-gram dan maksimal fitur	Mendapatkan *baseline*
2	Pengujian *Support Vector Machine* terhadap *hyperparameter tuning*	Mengetahui performa model dengan *hyperparameter tuning*
3	Pengujian *Support Vector Machine* terhadap ekspansi fitur	Mengetahui performa model dengan implementasi ekspansi fitur

a. Skenario Pertama (*Baseline*)

Skenario pertama dilakukan untuk mendapatkan *baseline* dengan mencari kombinasi pisah data, jumlah n-gram dan maksimal fitur pada TF-IDF yang digunakan pada data terhadap performa dari *Support Vector Machine* (SVM). Tabel 8 menjelaskan performa dari *Support Vector Machine* (SVM) pada skenario penelitian yang telah dilakukan. Akurasi terbesar didapat Ketika menggunakan skenario pisah data 70:30 dengan kombinasi n-gram adalah unigram dan bigram pada 5500 maksimal fitur dengan akurasi sebesar 76.82% yang akan digunakan sebagai *baseline* untuk tahap selanjutnya.

TABEL 8. Akurasi SVM Pada Skenario Percobaan 1

		Akurasi (%)		
Skenario Pisah Data	**ngram**	**4500**	**5000**	**5500**
90:10	Unigram	74.30	76.59	74.47
	Unigram + Bigram	74.47	74.13	73.78
	Unigram + Bigram + Trigram	73.78	74.30	73.95
80:20	Unigram	75.49	75.23	75.49
	Unigram + Bigram	76.45	76.28	76.19
	unigram,bigram,trigram	76.19	76.28	76.54
70:30	Unigram	76.36	76.59	76.76
	Unigram + Bigram	76.68	76.65	**76.82**
	Unigram + Bigram + Trigram	76.76	76.80	76.59

b. Skenario Kedua (*Hyperparameter Tuning*)

Skenario kedua dilakukan untuk mengetahui perbandingan performa dari *baseline* dengan *Support Vector Machine* (SVM) yang sudah dilakukan *hyperpatameter tuning*. Pada skenario pengujian kedua *hyperparamter tuning* pada parameter C, gamma dan fungsi kernel dilakukan dengan mencari kombinasi yang memiliki akurasi terbaik dari setiap nilai parameter yang dapat dilihat pada tabel 9.

TABEL 9. Skenario *Hyperparameter Tuning*

Parameter	Nilai yang diuji
Kernel	Rbf, poly, linear, sigmoid
C	0.1, 1, 10, 100, 1000
Gamma	1, 0.1, 0.01, 0.001, 0.0001

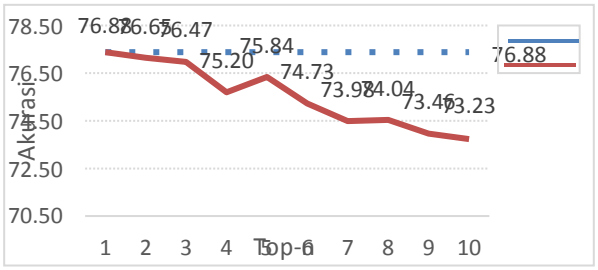
Hasil pengujian menunjukan performa dari *Support Vector Machine* (SVM) pada penelitian ini mengalami kenaikan akurasi menjadi 76.88% (+0.06%) setelah dilakukan *hyperparameter tuning* dengan kombinasi parameter C bernilai 10, gamma bernilai 1 dan fungsi kernel yang dipakai adalah RBF yang dapat dilihat pada tabel 10.

TABEL 10. Akurasi SVM Dengan *Hyperparameter Tuning*

Metode	Akurasi (%)
SVM	76.82
SVM + *Hyperparameter tuning*	**76.88 (+0,06)**

c. Skenario Ketiga (Ekspansi Fitur)

Skenario ketiga dilakukan dengan mengimplementasikan ekspansi fitur menggunakan vektor kata *pretrained* dari FastText pada metode *Support Vector Machine* (SVM). Pengujian ekspansi fitur dilakukan sebanyak 10 kali untuk *top-n* dengan n bernilai 1 sampai 10 dari kata yang berhubungan secara semantik dari vektor kata *pretrained* FastText. *Top-n* mengambil kata yang memiliki nilai kedekatan tertinggi sejumlah n. Hasil pengujian skenario ketiga dapat dilihat pada gambar 2. Dimana dapat dilihat bahwa sebagian besar akurasi dari metode *Support Vector Machine* (SVM) mengalami penurunan dan pada n bernilai 1 akurasi dari *Support Vector Machine* (SVM) tidak mengalami kenaikan maupun penurunan.

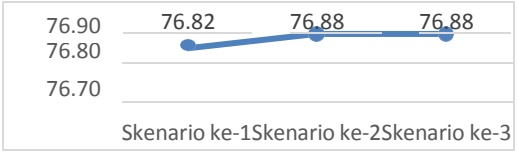


GAMBAR 2. Hasil Pengujian Ekspansi Fitur per Top-n

B. Analisis Hasil Pengujian

Dari ketiga skenario dapat diketahui bahwa tahapan yang dilakukan pada penelitian ini mempengaruhi performa yang dinilai. Akurasi yang didapatkan untuk metode pengklasifikasian *Support Vector Machine* (SVM) pada skenario pertama dapat diketahui bahwa *baseline* terbaik menggunakan skenario *split data* 70:30 dengan kombinasi unigram dan bigram dengan maksimal fitur sebesar 5500 yang mendapatkan akurasi sebesar 76.82% dan untuk data yang digunakan terdapat kenaikan dengan rata-rata sebesar 1.12% yang memperbesar data test, 0.01% menggunakan kombinasi unigram dan bigram dan 0.26% dengan maksimal fitur sebesar 5000. *Hyperparameter tuning* pada parameter c, gamma, dan fungsi kernel juga berpengaruh baik pada metode *Support Vector Machine* (SVM) dengan kenaikan akurasi sebesar 0.06% dari *baseline*. Penggunaan ekspansi fitur dengan vektor kata *pretrained* dari FastText pada top-2 hingga top-10 mengalami penurunan akurasi pada metode *Support Vector Machine* (SVM) sementara untuk penggunaan top 1 tidak mengalami kenaikan maupun penurunan pada akurasi dari metode *Support Vector Machine* (SVM).

Hasil skenario percobaan yang telah dilakukan pada bagian sebelumnya digambarkan pada gambar 2 yang menjelaskan akurasi terbaik dari skenario pertama hingga ketiga. Berdasarkan hasil pengujian pada setiap skenario pada gambar 3 menunjukkan adanya kenaikan sebesar 0.06% dari 76.82% (akurasi *baseline*) menjadi 78.88% saat dilakukan *hyperparameter tuning* namun tidak mengalami kenaikan akurasi dari *hyperparameter tuning* hingga implementasi ekspansi fitur pada metode *Support Vector Machine* (SVM).



GAMBAR 3. Akurasi Terbaik Setiap Skenario Percobaan

V. KESIMPULAN

Penelitian ini telah menjelaskan implementasi pencarian skenario split data, kombinasi n-gram dan jumlah maksimal fitur terbaik. H*yperparameter tuning* juga telah diterapkan untuk mendapatkan parameter dan kernel terbaik dari model klasifikasi *Support Vector Machine* (SVM) dan implementasi ekspansi fitur menggunakan model *pre-trained* berbahasa Indonesia dari FastText dengan metode pengklasifikasian *Support Vector Machine* (SVM).

Pada penelitian ini terbukti bahwa pemilihan skenario pisah data, n-gram dan maksimal fitur dapat mempengaruhi kinerja dari *Support Vector Machine* (SVM) untuk pencarian *baseline* terbaik dan mendapatkan akurasi sebesar 76.82% . Akurasi setelah implementasi *hyperparameter tuning* pada parameter C, gamma dan fungsi kernel juga terbukti dapat meningkatkan akurasi dari model klasifikasi menjadi 76.88% (+0.06%) dari *baseline*. Implementasi ekspansi fitur pada top 1 sampai 10 menggunakan vektor kata *pretrained* Bahasa Indonesia dari FastText membuat akurasi dari model klasifikasi *Support Vector Machine* (SVM) menurun, sedangkan pada top 1 akurasi tidak mengalami kenaikan maupun penurunan dibandingkan dengan hasil skenario percobaan sebelumnya 76.88% (*hyperparameter tuning*), maka dapat disimpulkan bahwa penggunaan ekspansi fitur dengan vektor kata *pretrained* Bahasa Indonesia tidak berpengaruh secara baik pada klasifikasi *tweet* berbahasa Indonesia menggunakan model *Support Vector Machine* (SVM).

REFERENSI

[1] S. Phuvipadawat and T. Murata, “Breaking news detection and tracking in Twitter,” *Proc. - 2010 IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol. - Work. WI-IAT 2010*, pp. 120–123, 2010, doi: 10.1109/WI-IAT.2010.205.

[2] K. Lee, D. Palsetia, R. Narayanan, M. M. A. Patwary, A. Agrawal, and A. Choudhary, “Twitter trending topic classification,” *Proc. - IEEE Int. Conf. Data Mining, ICDM*, pp. 251–258, 2011, doi: 10.1109/ICDMW.2011.171.

[3] E. B. Setiawan, D. H. Widyantoro, and K. Surendro, “Feature expansion using word embedding for tweet topic classification,” *Proceeding 2016 10th Int. Conf. Telecommun. Syst. Serv. Appl. TSSA 2016 Spec. Issue Radar Technol.*, no. 2011, 2017, doi: 10.1109/TSSA.2016.7871085.

[4] J. C. Young and A. Rusli, “Review and Visualization of Facebook’s FastText Pretrained Word Vector Model,” *2019 Int. Conf. Eng. Sci. Ind. Appl. ICESI 2019*, pp. 1–6, 2019, doi: 10.1109/ICESI.2019.8863015.

[5] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information,” *Trans. Assoc.*

Comput. Linguist., vol. 5, pp. 135–146, 2017, doi: 10.1162/tacl_a_00051.

[6] L. J. Chen and G. K. Hoon, "Feature expansion using lexical ontology for opinion type detection in tourism reviews domain," *Int. J. Adv. Comput. Sci. Appl.*, vol. 11, no. 8, pp. 628–637, 2020, doi: 10.14569/IJACSA.2020.0110877.

[7] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, "Learning word vectors for 157 languages," *Lr. 2018 - 11th Int. Conf. Lang. Resour. Eval.*, pp. 3483–3487, 2019.

[8] M. S. Saputri, R. Mahendra, and M. Adriani, "Emotion Classification on Indonesian Twitter Dataset," *Proc. 2018 Int. Conf. Asian Lang. Process. IALP 2018*, no. November, pp. 90–95, 2019, doi: 10.1109/IALP.2018.8629262.

[9] N. Yusliani, R. Primartha, and M. Diana, "Multiprocessing Stemming: A Case Study of Indonesian Stemming," *Int. J. Comput. Appl.*, vol. 182, no. 40, pp. 15–19, 2019, doi: 10.5120/ijca2019918476.

[10] B. Y. Pratama and R. Sarno, "Personality classification based on Twitter text using Naive Bayes, KNN and SVM," *Proc. 2015 Int. Conf. Data Softw. Eng. ICODSE 2015*, no. November, pp. 170–174, 2016, doi: 10.1109/ICODSE.2015.7436992.

[11] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations ofwords and phrases and their compositionality," *Adv. Neural Inf. Process. Syst.*, no. October 2013, 2013.

[12] A. Basarkar, "Document Classification Using Machine Learning," vol. IX, no. Vi, pp. 60–67, 1998.

[13] M. . Imelda A.Muis & Muhammad Affandes, "Penerapan Metode Support Vector Machine (SVM) Menggunakan Kernel Radial Basis Function (RBF) Pada Klasifikasi Tweet," *Sains, Teknol. dan Ind. Sultan Syarif Kasim Riau*, vol. 12, no. 2, pp. 189–197, 2015.

[14] M. H. Afif and A. R. Hedar, "Data classification using support vector machine integrated with scatter search method," *Proc. 2012 Japan-Egypt Conf. Electron. Commun. Comput. JEC-ECC 2012*, pp. 168–172, 2012, doi: 10.1109/JEC-ECC.2012.6186977.

[15] F. Pedegrosa *et al.*, "Scikit-learn: Machine Learning in Python," *Mach. Learn. Res.*, vol. 12, no. 1, pp. 2825–2830, 2018.