Analisis Sentimen Twitter Bahasa Indonesia dengan *Word2Vec*

Farhan Wahyu Kurniawan[1], Dr. Warih Maharani, S.T., M.T.[2]

[1,2]Fakultas Informatika, Universitas Telkom, Bandung
[1]farhanwk@student.telkomuniversity.ac.id, [2]wmaharani@telkomuniversity.ac.id

Abstrak

Sentimen merupakan opini seseorang terhadap suatu topik, produk, atau layanan tertentu. Analisis sentimen digunakan untuk menganalisis opini terhadap suatu topik tertentu apakah cenderung positif atau negatif. Media sosial Twitter digunakan oleh masyarakat Indonesia untuk menuliskan opini mereka dalam bentuk cuitan (*tweet)*. Penelitian ini menjelaskan klasifikasi sentimen pada data Twitter berbahasa Indonesia untuk membantu dalam memahami sentimen pengguna Indonesia terhadap suatu topik yang dibahas di Twitter. Penelitian ini menggunakan metode *word2vec* untuk mengekstraksi fitur dengan mengkonversi data menjadi nilai vektor. *Word2Vec* memiliki keunggulan dapat melihat hubungan semantik antar kata. Metode klasifikasi pada penelitian ini menggunakan *support vector machine* (SVM). Proses klasifikasi sentimen dilakukan dengan mengolah data latih berupa data cuitan yang sudah dikumpulkan yang kemudian menjadi model untuk proses pengujian pada data uji. Dari hasil pengujian, penerapan metode *word2vec* dan SVM menghasilkan tingkat presisi sebesar 64,4%, *recall* sebesar 58%, dan *f-score* sebesar 61,1%

Kata kunci : analisis sentimen, twitter, *word2vec*, svm

Abstract

Sentiment is an opinion of someone on a certain topics, products or services. Sentiment Analysis is used to analyze opinion on a certain topics to decide whether they are positive or negative. Twitter is used by Indonesian to express their opinion in the form of tweets. This study used word2vec method to extract features by converting data into vector score. Word2Vec has the advantage of being able to see semantic relationships between words. This study used support vector machine (SVM) classification method. The sentiment classification process is done by processing training data in the form of collected tweets then turned into model to be tested with testing data. This result of applying word2vec and SVM produce score of precision 64,4%, recall 58%, and f-score 61,1%.

Keywords: sentiment analysis, twitter, word2vec, svm

1. Pendahuluan

Latar Belakang

Penggunaan media sosial sudah semakin populer pada saat ini. Media sosial telah menjadi tempat untuk para pengguna bertukar pikiran terkait isu atau topik yang bermacam-macam seperti olahraga, politik, keuangan, gaya hidup, dan lain sebagainya. Twitter merupakan salah satu media sosial yang dipakai oleh masyarakat Indonesia. Menurut data dari statista, jumlah pengguna Twitter di Indonesia mencapai 22,8 juta pengguna[1]. Pengguna menuliskan opini mereka dalam bentuk cuitan yang ditampilkan secara publik sehingga setiap orang dapat melihat opini tersebut. Opini publik telah banyak digunakan untuk memahami sentimen terhadap topik, produk atau seseorang [1]. Sentimen diklasifikasikan ke dalam ketegori positif atau negatif. Proses menganalisis sentimen terhadap suatu produk, layanan, organisasi atau individu disebut analisis sentimen [2].

Pada analisis sentimen terdapat tahap ekstraksi fitur. Ekstraksi fitur merupakan tahap di mana kata yang dapat menjelaskan sentimen pada *dataset* diekstraksi menjadi fitur atau aspek [3]. TF-IDF merupakan salah satu metode yang sering digunakan untuk ekstraksi fitur [4]. TF-IDF memiliki keunggulan yaitu mudah diimplementasikan. Namun TF-IDF memiliki kelemahan yaitu tidak dapat memproses relasi semantik antar kata sehingga menganggap setiap kata memiliki konteks yang berbeda [5]. Seperti kata *'car'* dan *'automobile'* atau *'buy'* dengan *'buys'* yang memiliki konteks yang sama namun dianggap tidak memiliki kedekatan konteks. Pada analisis sentimen, relasi semantik memberikan dampak berupa hasil klasifikasi yang lebih baik [6]. Ekstraksi fitur dengan relasi semantik dapat dilakukan dengan metode *word embedding*. *Word2Vec* merupakan salah satu metode *word embedding* di mana *word2vec* mempelajari representasi kata pada ruang vektor dengan dimensi yang tinggi[2]. *Word2Vec* menghitung *cosine similarity* antar kata melalui nilai vektor kata sehingga diketahui hubungan semantik antar kata [6]. Pada penelitian sebelumnya dibandingkan penerapan *word2vec* dan TF-IDF untuk analisis sentimen [1]. Dari penelitian tersebut didapatkan bahwa klasifikasi dengan menggunakan

[1] https://www/statista.com
[2] https://code.google.com

word2vec sebagai ekstraksi fitur menghasilkan akurasi yang lebih baik daripada menggunakan TF-IDF. Sehingga pada penelitian ini akan digunakan *word2vec* untuk ekstraksi fitur.

Setelah ekstraksi fitur, dilakukan proses klasifikasi sentimen. Pada penelitian ini data diklasifikasi menggunakan *support vector machine* (SVM). SVM merupakan metode yang paling akurat untuk klasifikasi dibandingkan dengan metode lain seperti *naive bayes*, dan *decision tree* [7]. Pada penelitian sebelumnya, *word2vec* dan SVM digunakan untuk klasifikasi sentimen dan menghasilkan akurasi yang baik [1]. Namun dataset pada penelitian tersebut berupa cuitan berbahasa Inggris, bukan berbahasa Indonesia. Sehingga pada penelitian ini akan dilakukan percobaan dengan metode yang sama dan *dataset* yang berbeda, yaitu berupa cuitan berbahasa Indonesia.

Permasalahan

Permasalahan pada penelitian ini yaitu bagaimana penerapan *word2vec* yang tepat sebagai model ekstraksi fitur terhadap analisis sentimen Twitter berbahasa Indonesia dengan menggunakan SVM sebagai metode klasifikasi. Metode diimplementasikan dengan menggunakan bahasa *python*. Data yang dipakai berupa cuitan (*tweet*) dengan tema bencana di Indonesia.

Tujuan

Tujuan dari penelitian ini yaitu menganalisis faktor-faktor yang mempengaruhi hasil penerapan metode *word2vec* seperti jenis arsitektur model dan ukuran dimensi pada analisis sentimen Twitter berbahasa Indonesia. Hasil penerapan akan dilihat dari skor presisi*, recall dan f-score* [8].

2. Studi Terkait

2.1. Analisis Sentimen Twitter

Twitter merepresentasikan dirinya sebagi *platform* yang berisi tentang apa yang sedang terjadi di dunia dan apa yang sedang orang bicarakan saat ini[3]. Twitter adalah sebuah layanan untuk berkomunikasi melalui pertukaran cuitan (*tweet*). Cuitan (*tweet*) dapat berisi foto, video, tautan, dan teks. Cuitan (*tweet*) yang dibuat oleh seorang pengguna akan dikirim ke pengikutnya[4]. Analisis sentimen merupakan proses mengidentifikasi opini atau sentimen dari suatu entitas [2]. Entitas pada analisis sentimen seperti produk, layanan, seseorang, atau sebuah organisasi. Proses mengidentifikasi sentimen yaitu dengan mengekspresikan kecenderungan opini dari sebuah data atau dokumen ke arah positif atau negatif. Penelitian analisis sentimen mulai ramai pada tahun 2000. Hal tersebut dikarenakan analisis sentimen mulai diaplikasikan ke berbagai hal seperti bisnis dan politik serta berlimpahnya data yang tersedia dengan mulai ramainya penggunaan media sosial [2].

2.2. *Pre-processing*

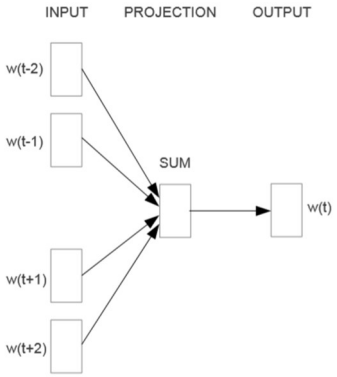
Pre-processing adalah proses menghilangkan *noise* dari data [9]. Data teks mengandung karakter atau kata yang tidak relevan pada proses klasifikasi sehingga dieleminasi agar proses klasifikasi lebih efisien dan akurat [1]. Data teks yang bersumber dari Twitter juga tidak berbeda dengan data teks pada umumnya, yaitu bersifat *noisy* dan tidak terstruktur [9]. Cuitan (*tweet*) di Twitter mengandung karakter atau simbol khusus seperti *mention* (*@username*), *retweet* (RT), *URL*, dan *hashtag* (#). Maka dari itu, proses *pre-processing* perlu dilakukan sebelum proses klasifikasi sentimen.

2.3. *Word2Vec*

Word2Vec didasarkan pada ide *deep learning* di mana kata direpresentasikan dalam vektor [10]. *Word2Vec* mentrasformasikan operasi dokumen menjadi perhitungan vektor dalam ruang vektor kata. Relasi semantik pada dokumen dapat dikarakterisasi berdasarkan kesamaan kata di dalam ruang vektor. Tahap awal pada proses *word2vec* yaitu membangun kosakata dari data teks pelatihan dan kemudian mempelajari representasi vektor dari kumpulan kata[5]. Vektor yang dihasilkan dapat digunakan sebagai fitur untuk penerapan dalam kasus *natural language processing* dan *machine learning*. *Word2Vec* terdiri dari dua arsitektur, yaitu:

[3] https://help.twitter.com
[4] https://help.twitter.com
[5] https://code.google.com

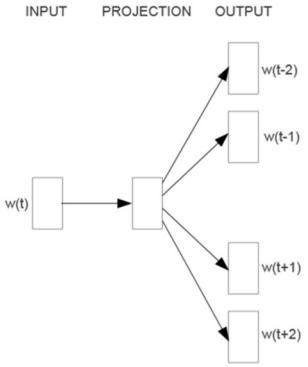
- ***Continous Bags-of-Words*** **(CBOW)**



Gambar 2.1 Model arsitektur *continous bags-of-words* (CBOW) [11]

CBOW merupakan sebuah model arsitektur yang didasarkan pada *neural network* di mana *non-linear hidden layer* dihilangkan dan *projection layer* dibagikan ke semua kata [11]. CBOW memprediksi suatu kata berdasarkan kata-kata lain yang berada dalam satu kalimat. Tidak seperti model *bag-of-words*, pada model ini digunakan representasi yang terdistribusi secara kontinyu.

- ***Skip-gram***



Gambar 2.2 Model arsitektur *skip-gram* [11]

Model arsitektur *skip-gram* mirip seperti CBOW. Pelatihan dari model *skip-gram* yaitu untuk representasi kata yang berguna dalam memprediksi kata-kata yang ada di dekatnya pada suatu kalimat atau dokumen [12]. *Skip-gram* melakukan proses prediksi dengan melihat kata yang ada di sebelum dan sesudah *current word* [11]. Model *skip-gram* memaksimalkan rata-rata *log probability* yang dapat dilihat pada persamaan berikut.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{2.1}$$

w_t = kata *center*
w_{t+j} = kata setelah kata *center*
c = ukuran *training context*

Nilai vektor dokumen atau dalam penelitian ini merupakan nilai vektor cuitan (*tweet*). Nilai vektor dokumen didapatkan dengan menghitung nilai rata-rata vektor dari semua kata di dalam satu cuitan (*tweet*) [10]. Berikut persamaan untuk menghitung nilai vektor dokumen.

$$vec_{c,j} = \frac{1}{n_j}\sum_{i=1}^{m}\omega_i V(w_i) \tag{2.2}$$

$vec_{c,j}$ = vektor dokumen j pada *class* c
n_j = jumlah fitur dalam dokumen j
$V(w_i)$ = vektor kata dari fitur w
ω_i = nilai vektor dari fitur w

2.4. *Support Vector Machine* (SVM)

Support Vector Machine merupakan salah satu klasifikasi *supervised.* Proses kerja SVM yaitu dengan mencoba memberikan garis batas kepada dua buah kategori, atau yang biasa disebut *hyperplane* [7]. *Hyperplane*

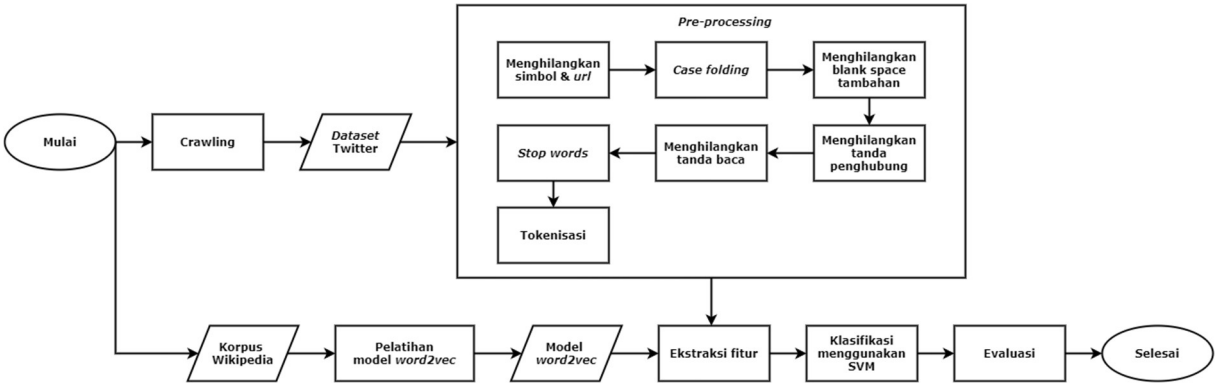
dibentuk dengan melihat data yang paling dengan dengan *hyperplane*, atau biasa disebut *support vector*. Pada penelitian ini menggunakan *kernel linear* dengan persamaan sebagai berikut.

$$w.x + b = 0 \quad (2.3)$$

Di mana w merupakan vektor *weight* dan x adalah vektor dari atribut *dataset*, sedangkan b adalah nilai bias. SVM menentukan *hyperplane* untuk memaksimalkan *margin* dan juga mengurangi misklasifikasi [7].

3. Sistem yang Dibangun

Pada penelitian ini akan dibangun model yang mengekstraksi fitur dan mengklasifikasikan data teks berupa cuitan (*tweet*) dari Twitter ke dalam sentimen positif atau negatif. Ekstraksi fitur akan menggunakan metode *word2vec* dan untuk klasifikasi sentimen akan menggunakan metode *support vector mechine* (SVM). Alur pemodelan pada penelitian ini ditunjukan pada gambar 3.1.



Gambar 3.1 Gambaran umum rancangan sistem

3.1. *Crawling*

Crawling merupakan proses pengambilan data dari sumber data yang nantinya akan menjadi *dataset*. Pada penelitian ini data yang berupa cuitan (*tweet*) diambil dari situs media sosial Twitter. Data diambil dengan menentukan kata kunci untuk pencarian cuitan (*tweet*). Proses *crawling* dilakukan dengan memanfaatkan API yang disediakan oleh Twitter[6]. Tahap awal untuk mendapatkan akses ke data Twitter yaitu dengan mendaftarkan akun sebagai *developer* dan mendeskripsikan alasan kenapa ingin mendapatkan akses API, yang dalam hal ini yaitu untuk penelitian akademik. Setelah pendaftaran selesai, Twitter akan memberikan 4 *secret key: consumer_key; consumer_secret; access_token;* dan *access_token_secret. Secret key* tersebut akan digunakan untuk kunci mengakses API Twitter. Proses *crawling* menggunakan bahasa pemrograman *python* 3. Algortima untuk proses *crawling* sebagai berikut.

```
Begin
      Insialisasi secret key
      Inisialisasi file CSV sebagai tempat menyimpan dataset
      Inisialisasi kata kunci yang ingin di-crawl
      Melakukan proses autentifikasi dengan secret key ke server Twitter
      Search cuitan berdasarkan kata kunci
      Write di CSV
End
```

Proses *crawling* menghasilkan *username* dan cuitan (*tweet*) yang disimpan pada *file* CSV. Contoh cuitan (*tweet*) yang tersimpan seperti berikut.

Tabel 3.1 Contoh hasil *crawling*

Username	Cuitan
@usaidindonesia	*"Anak-anak muda jadi ujung tombak peningkatan sadar #bencana. Ipul dari Kota #Ambon bantu sebar informasi untuk turunkan risiko bencana!. Pc J Hattu @USAID APIK http://ow.ly/ZbTB50yf0jH"*

3.2. *Pre-processing*

Dataset yang sudah di-*crawl* masih dalam bentuk yang tidak terstruktur dan *noisy.* Masih terdapat tulisan-tulisan yang tidak diperlukan untuk proses klasifikasi sentimen seperti *url, mention, hashtag*, dll. *Dataset* tersebut perlu melalui proses pembersihan teks. Pada proses ini terdapat tahapan *stop words* yang menggunakan

[6] https://developer.twitter.com

library Sastrawi dengan daftar kata *default*[7]. Proses membersihkan data dari noise disebut *pre-processing* [9]. Tahapan *pre-processing* pada penelitian ini yaitu:

Tabel 3.2 Tahapan *pre-processing*

Tahapan	Hasil
Sebelum dilakukan *pre-processing*	*'Anak-anak muda jadi ujung tombak peningkatan sadar #bencana. Ipul dari Kota #Ambon bantu sebar informasi untuk turunkan risiko bencana!. Pc J Hattu @USAID APIK http://ow.ly/ZbTB50yf0jH*
Menghilangkan simbol dan *url*	*Anak-anak muda jadi ujung tombak peningkatan sadar. Ipul dari Kota bantu sebar informasi untuk turunkan risiko bencana!. Pc J Hattu*
Case folding	*anak-anak muda jadi ujung tombak peningkatan sadar. ipul dari kota bantu sebar informasi untuk turunkan risiko bencana!. pc j hattu*
Menghilangkan ekstra *blank space*	*anak-anak muda jadi ujung tombak peningkatan sadar. ipul dari kota bantu sebar informasi untuk turunkan risiko bencana!. pc j hattu*
Menghilangkan tanda penghubung	*anak anak muda jadi ujung tombak peningkatan sadar. ipul dari kota bantu sebar informasi untuk turunkan risiko bencana!. pc j hattu*
Menghilangkan tanda baca	*anak anak muda jadi ujung tombak peningkatan sadar ipul dari kota bantu sebar informasi untuk turunkan risiko bencana pc j hattu*
Stop words	*anak anak muda ujung tombak peningkatan sadar ipul kota bantu sebar informasi turunkan risiko bencana pc j hattu*
Tokenisasi	*[anak, anak, muda, ujung, tombak, peningkatan, sadar, ipul, kota, bantu, sebar, informasi, turunkan, risiko, bencana, pc, j, hattu]*

3.3. Pelatihan *word2vec*

Pada tahap ini akan dilatih data korpus Wikipedia berbahasa Indonesia dengan metode word2vec. Pelatihan word2vec memerlukan data dengan jumlah kata yang lengkap, sehingga pada penelitian ini menggunakan data korpus Wikipedia. Output dari pelatihan ini yaitu model *word2vec* yang digunakan untuk mengekstraksi fitur dari dataset. *Word2Vec* menggunakan arsitektur CBOW dan *skip-gram* untuk melakukan proses pelatihan data.

3.3.1. Contoh pelatihan dengan arsitektur CBOW

Pada contoh ini digunakan satu kalimat cuitan *"banjir menerjang beberapa wilayah jakarta"*. Kalimat tersebut direpresentasikan terlebih dahulu dalam bentuk *one-hot encoding*. Kata diubah menjadi bentuk kumpulan angka dalam matriks. Untuk kata *"banjir"* diubah menjadi $[1,0,0,0,0]^T$, dan untuk kata *"menerjang"* diubah menjadi $[0,1,0,0,0]^T$. Pada contoh ini akan diprediksi kata *"menerjang"* dengan input kata *"banjir"* dan *"beberapa"*. Prediksi menggunakan arsitektur CBOW dengan *window context* C=1, jumlah kata V=5 dan jumlah dimensi N=3. Nilai matriks *input* dan *output* diasumsikan untuk contoh kasus ini. Berikut alur perhitungan CBOW sampai mendapatkan nilai *loss*.

$$\underset{(Vx1)}{X_1} \quad \underset{(VxN)}{W_{input}} \quad \underset{(Nx1)}{h_{x1}}$$

$$\begin{bmatrix}1\\0\\0\\0\\0\end{bmatrix} x \begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\\10 & 11 & 12\\13 & 14 & 15\end{bmatrix}^T = \begin{bmatrix}1\\2\\3\end{bmatrix}$$

$$\underset{(Vx1)}{X_2} \quad \underset{(VxN)}{W_{input}} \quad \underset{(Nx1)}{h_{x2}}$$

$$\begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix} x \begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\\10 & 11 & 12\\13 & 14 & 15\end{bmatrix}^T = \begin{bmatrix}7\\8\\9\end{bmatrix}$$

$$\underset{(NxV)}{W_{output}} \quad \underset{(Nx1)}{h_{avg}} \quad \underset{(Vx1)}{W^T_{output}.h_{avg}} \quad \underset{(Vx1)}{y_{pred}}$$

$$\begin{bmatrix}0{,}11 & 0{,}12 & 0{,}13 & 0{,}14 & 0{,}15\\0{,}16 & 0{,}17 & 0{,}18 & 0{,}19 & 0{,}2\\0{,}21 & 0{,}22 & 0{,}23 & 0{,}24 & 0{,}25\end{bmatrix}^T x \begin{bmatrix}4\\5\\6\end{bmatrix} = \begin{bmatrix}2{,}5\\2{,}65\\2{,}8\\2{,}95\\3{,}1\end{bmatrix} \rightarrow \begin{bmatrix}0{,}145\\0{,}168\\0{,}196\\0{,}227\\0{,}264\end{bmatrix}$$

$$\underset{(Vx1)}{y_{pred}} \quad \underset{(Vx1)}{y_{target}} \quad \underset{(Vx1)}{loss}$$

$$\begin{bmatrix}0\\1\\0\\0\\0\end{bmatrix} - \begin{bmatrix}0{,}145\\0{,}168\\0{,}196\\0{,}227\\0{,}264\end{bmatrix} = \begin{bmatrix}-0{,}145\\0{,}832\\-0{,}196\\-0{,}227\\-0{,}264\end{bmatrix}$$

[7] github.com/har07/PySastrawi

3.3.2. Contoh pelatihan dengan arsitektur *skip-gram*

Pada contoh *skip-gram*, kalimat yang digunakan sama seperti pada contoh pelatihan CBOW. *Input* kata pada pelatihan ini yaitu *"menerjang"* yang memiliki matriks $[0,1,0,0,0]^T$. Kata *"menerjang"* akan digunakan untuk memprediksi kata *"banjir"* dan *"beberapa"* yang memiliki matriks $[1,0,0,0,0]^T$ dan $[0,0,1,0,0]^T$. Spesifikasi yang digunakan seperti *window context, jumlah kata,* dan jumlah dimensi sama seperti pada pelatihan CBOW. Berikut alur perhitungan skip-gram sampai mendapatkan nilai *loss*.

$$\underset{(Vx1)}{X} \quad \underset{(VxN)}{W_{input}} \quad \underset{(Nx1)}{h}$$

$$\begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix} x \begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\\10 & 11 & 12\\13 & 14 & 15\end{bmatrix}^T = \begin{bmatrix}4\\5\\6\end{bmatrix}$$

$$\underset{(NxV)}{W_{output}} \quad \underset{(Nx1)}{h} \quad \underset{(Vx1)}{W^T_{output}.h} \quad \underset{(Vx1)}{y_{pred}}$$

$$\begin{bmatrix}0{,}11 & 0{,}12 & 0{,}13 & 0{,}14 & 0{,}15\\0{,}16 & 0{,}17 & 0{,}18 & 0{,}19 & 0{,}2\\0{,}21 & 0{,}22 & 0{,}23 & 0{,}24 & 0{,}25\end{bmatrix}^T x \begin{bmatrix}4\\5\\6\end{bmatrix} = \begin{bmatrix}2{,}5\\2{,}65\\2{,}8\\2{,}95\\3{,}1\end{bmatrix} \rightarrow \begin{bmatrix}0{,}145\\0{,}168\\0{,}196\\0{,}227\\0{,}264\end{bmatrix}$$

$$\underset{(Vx1)}{y_{pred}} \quad \underset{(Vx1)}{y_{target\ w+1}} \quad \underset{(Vx1)}{loss_{w+1}} \qquad \underset{(Vx1)}{y_{pred}} \quad \underset{(Vx1)}{y_{target\ w-1}} \quad \underset{(Vx1)}{loss_{w-1}}$$

$$\begin{bmatrix}1\\0\\0\\0\\0\end{bmatrix} - \begin{bmatrix}0{,}145\\0{,}168\\0{,}196\\0{,}227\\0{,}264\end{bmatrix} = \begin{bmatrix}0{,}855\\-0{,}168\\-0{,}196\\-0{,}227\\-0{,}264\end{bmatrix} \qquad \begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix} - \begin{bmatrix}0{,}145\\0{,}168\\0{,}196\\0{,}227\\0{,}264\end{bmatrix} = \begin{bmatrix}-0{,}145\\-0{,}168\\0{,}804\\-0{,}227\\-0{,}264\end{bmatrix}$$

$$\underset{(Vx1)}{loss_{w+1}} \quad \underset{(Vx1)}{loss_{w-1}} \quad \underset{(Vx1)}{loss}$$

$$\begin{bmatrix}0{,}855\\-0{,}168\\-0{,}196\\-0{,}227\\-0{,}264\end{bmatrix} + \begin{bmatrix}-0{,}145\\-0{,}168\\0{,}804\\-0{,}227\\-0{,}264\end{bmatrix} = \begin{bmatrix}0.710\\-0{,}337\\0{,}609\\-0{,}454\\-0{,}528\end{bmatrix}$$

Pada proses ini, *word2vec* akan mempelajari semua kata pada *dataset* dengan output berupa representasi vektor untuk setiap kata dalam ruang vektor yang berdimensi tinggi. Nilai *loss/error* akan digunakan untuk memperbarui *weight* pada *input* dan *output weight matrix* pada tahap *back propagation*. Pada model word2vec dapat dilihat kedekatan antar kata seperti kata "bagus" yang mempunyai kedekatan dengan kata "mumpuni", "menggembirakan", "memuaskan", "mengesankan", "mengagumkan", "baik", "buruk", "jelek", "impresif", "cemerlang".

3.4. Ekstraksi fitur

Setelah *dataset* dibersihkan di tahap *pre-processing*, selanjutnya yaitu mengekstraksi fitur. Setiap kata pada *dataset* dilihat nilai vektornya pada model *word2vec* yang sudah dilatih. Nilai vektor dokumen atau cuitan didapatkan dengan menghitung nilai rata-rata vektor dari semua kata di dalam satu cuitan. Nilai vektor dokumen digunakan sebagai *input* untuk proses klasifikasi.

3.5. Klasifikasi menggunakan SVM

SVM merupakan *supervised classifier,* sehingga dataset perlu dibagi dua untuk proses pelatihan dan pengujian. Data untuk proses pelatihan diberi label terlebih dahulu. Label untuk klasifikasi sentimen terdiri dari positif dan negatif. Berikut contoh pelabelan pada data.

Tabel 3.3 Contoh pelabelan data

Cuitan	Label
Masihkah kita menyalahkan Hujan? Menyalahkan Hujan Berarti menyalahkan Tuhan	Negatif
Sedia tanaman sebelum hujan. Perilaku ini akan menjaga kita dari bencana	Positif
Mending duit buat ngebangun terowongannye dipake dulu buat nanggulangin #bencana yg masih banyak kejadiannye di seluruh Tanah Air!!!	Negatif
Benar bahwa solusi pembangunan infrastruktur itu penting. Tapi selama ekologinya tidak diperbaiki, selama tidak dilakukan penanaman pohon, bencana tanah longsor akan terus terjadi	Negatif

Setelah proses pelabelan data, dilakukan proses pelatihan data yang mempelajari setiap kalimat dengan label yang sudah diberikan. SVM akan mencoba membuat garis *hyperplane* untuk memisahkan kalimat yang

memiliki sentimen positif dan negatif. *Margin* dari garis *hyperplane* akan dimaksimalkan dengan mengidentifikasi *support vector.* Hasil klasifikasi akan berupa model yang dapat digunakan untuk memprediksi sentimen pada data pengujian. Klasifikasi menggunakan library Sklearn SVC dengan parameter default di mana nilai C=1 dan *kernel*="*linear*". Proses selanjutnya yaitu memprediksi sentimen dari data uji berdasarkan model yang sudah dilatih dengan data latih.

3.6. Evaluasi hasil klasifikasi

Analisis sentimen merupakan salah satu bentuk klasifikasi, sehingga hasil dari proses klasifikasi harus dievaluasi. Cara yang biasa dilakukan untuk mengevaluasi yaitu dengan evaluasi matriks [8]. Evaluasi matriks terdiri dari presisi, *recall*, dan *f-score*. Hasil dari klasifikasi dibagi menjadi empat jenis, yaitu *True Positives* (TP), *True Negatives* (TN), *False Positives* (FP), and *False Negatives* (FN). Keempat jenis hasil klasifikasi digunakan untuk menentukan presisi, *recall*, dan *f-score* seperti pada persamaan berikut [8].

$$Presisi = \frac{TP}{TP + FP} \quad (3.1) \qquad Recall = \frac{TP}{TP + FN} \quad (3.2) \qquad F - score = 2\,x\,\frac{Presisi\ x\ Recall}{Presisi + Recall} \quad (3.3)$$

4. Evaluasi

Data yang digunakan pada penelitian ini berupa cuitan yang diambil pada media sosial Twitter dengan tema seputar bencana. Data yang terkumpul sebanyak 1110 cuitan. Data dibagi menjadi data latih sebanyak 1010 cuitan dan data uji sebanyak 100 cuitan. Korpus yang digunakan untuk membangun model *word2vec* menggunakan artikel Wikipedia berbahasa Indonesia versi 2 Februari 2020. Pengujian dilakukan dengan melihat hasil presisi, *recall*, dan *f-score* pada klasifikasi sentimen Twitter berbahasa Indonesia.

4.1. Analisis penggunaan model CBOW dan *skip-gram*

Pengujian dilakukan untuk melihat pengaruh perbedaan arsitektur dan ukuran dimensi model *word2vec* terhadap klasifikasi sentimen. Arsitektur model yang diuji yaitu CBOW dan *skip-gram* dengan ukuran dimensi 100, 200, dan 300 dimensi. Hasil dari pengujian dilihat dari perbandingan presisi, *recall*, dan *f-score.*

Tabel 4.1 Hasil perbandingan pengaruh dimensi terhadap klasifikasi

Model	Presisi	*Recall*	*F-score*
CBOW 100 dimensi	59,1%	52%	55,3%
CBOW 200 dimensi	59,5%	44%	50,6%
CBOW 300 dimensi	58,5%	48%	52,7%
Skip-gram 100 dimensi	64,4%	58%	61,1%
Skip-gram 200 dimensi	53,5%	46%	49,5%
Skip-gram 300 dimensi	60%	54%	56,8%

Berdasarkan pengujian didapatkan bahwa perbedaan arsitektur dan jumlah dimensi mempengaruhi hasil klasifikasi. Pada tabel 4.1 dapat dilihat bahwa model *skip-gram* 100 dimensi memberikan hasil klasifikasi paling baik. Model *skip-gram* 100 dimensi menghasilkan presisi sebesar 64,4%, *recall* sebesar 58%, dan *f-score* sebesar 61,1%. Hal ini dapat disebabkan karena perbedaan model menghasilkan nilai kedekatan kata yang berbeda-beda sehingga terdapat perbedaan nilai vektor untuk kata yang sama. Nilai vektor yang berbeda mempengaruhi data *input* untuk proses klasifikasi.

4.2. Analisis kamus kata pada model

Pengujian dilakukan untuk melihat jumlah dari kata yang terdapat dan tidak terdapat pada model. Kata yang tidak terdapat di model dikumpulkan dan dihitung ketika proses ekstraksi fitur. Beberapa cuitan diketahui terdiri dari semua kata yang tidak terdapat pada model sehingga cuitan diabaikan dan mengurangi jumlah keseluruhan data.

Tabel 4.2 Jumlah kata pada model

Deskripsi	Jumlah
Jumlah kata yang terdapat di model	3905
Jumlah kata yang tidak terdapat di model	577

Berdasarkan pengujian didapatkan bahwa ada kata yang tidak terdapat pada model. Pada tabel 4.2 dapat dilihat terdapat 577 kata yang tidak ada pada model sehingga tidak dapat dipakai. Hal tersebut dapat disebabkan karena kata tidak dalam bentuk baku atau tidak dengan cara penulisan yang benar sehingga kata tidak dikenali. Contoh kata yang tidak terdapat pada model seperti *"bnjr", "siagabencana"*. Kata yang tidak dipakai tidak dianggap dalam suatu kalimat sehingga hanya dihitung kata yang terdapat pada model. Pengaruh dari kata yang tidak dipakai yaitu mengurangi variasi kata untuk proses klasifikasi.

4.3. Analisis komposisi data

Pengujian komposisi data dilakukan untuk melihat pengaruh komposisi data terhadap klasifikasi. Pengujian dilakukan dengan mengubah jumlah data latih yang dipakai untuk pembuatan model klasifikasi. Hasil dari pengujian dilihat dari perbandingan presisi, *recall*, dan *f-score.*

Tabel 4.3 Hasil perbandingan komposisi data

Jumlah data latih	Presisi	Recall	F-score
210	53,8%	56%	54,9%
510	59,5%	50%	54,3%
1010	64,4%	58%	61,1%

Berdasarkan pengujian didapatkan bahwa jumlah data latih mempengaruhi hasil klasifikasi. Pada tabel 4.3 dapat dilihat bahwa komposisi data paling baik ketika menggunakan jumlah data latih sebanyak 1010 data. Penggunaan 1010 data latih menghasilkan tingkat presisi sebesar 64,4%, tingkat recall sebesar 58%, dan nilai f-score sebesar 61,1%. Hal tersebut dapat disebabkan karena data yang lebih banyak meningkatkan variasi data dengan jumlah kata yang semakin banyak. Variasi data yang lebih tinggi memberikan hasil klasifikasi yang lebih baik.

5. Kesimpulan

Dari pengujian ini dilakukan klasifikasi sentimen menggunakan model *word2vec* untuk estraksi fitur dan SVM untuk klasifikasi. Hasil dari pelatihan model *word2vec* didapatkan bahwa perbedaan jenis arsitektur model dan ukuran dimensi *word2vec* mempengaruhi hasil klasifikasi. Model *skip-gram* 100 dimensi memberikan hasil klasifikasi paling baik dengan tingkat presisi sebesar 64,4%, *recall* sebesar 58%, dan *f-score* sebesar 61,1%. Hasil klasifikasi paling baik didapatkan ketika menggunakan data latih sebanyak 1010 data. Pada pengujian ini juga diketahui ada kata yang tidak terdapat pada model sebanyak 577 kata. Saran untuk penelitian selanjutnya dapat menggunakan jumlah data yang lebih banyak sehingga meningkatkan variasi data. Selain itu, dapat menggunakan daftar *stopwords* yang lebih sesuai dengan data yang dipakai.

Daftar Pustaka

[1] M. K. Sohrabi and F. Hemmatian, "An efficient preprocessing method for supervised sentiment analysis by converting sentences to numerical vectors: a twitter case study," *Multimedia Tools and Applications,* pp. 1-20, 2019.

[2] B. Liu, Sentiment Analysis and Opinion Mining: Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers, 2012.

[3] M. Z. Asghar, A. Khan, S. Ahmad and F. M. Kundi, "A review of feature extraction in sentiment analysis," *Journal of Basic and Applied Scientific Research,* pp. 181-186, 2014.

[4] G. Paltoglou and M. Thelwall, " A study of information retrieval weighting schemes for sentiment analysis," *Proceedings of the 48th annual meeting of the association for computational linguistics,* pp. 1386-1395, 2010.

[5] J. Ramos, "Using tf-idf to determine word relevance in document queries," *Proceedings of the first instructional conference on machine learning,* pp. 133-142, 2003.

[6] D. Zhang, H. Xu, Z. Su and Y. Xu, "Chinese comments sentiment classification based on word2vec and svmperf," *Expert Systems with Applications,* pp. 1857-1863, 2015.

[7] H. Bhavsar and A. Ganatra, " A comparative study of training algorithms for supervised machine learning," *International Journal of Soft Computing and Engineering (IJSCE),* pp. 2231-2307, 2012.

[8] A. Giachanou and F. Crestani, " Like it or not: A survey of twitter sentiment analysis methods," *ACM Computing Surveys (CSUR),* 2016.

[9] A. F. Hidayatullah and M. R. Ma'arif, "Pre-processing tasks in Indonesian Twitter messages," *Journal of Physics: Conference Series,* 2017.

[10] W. Tian, J. Li and H. Li, "A method of feature selection based on word2vec in text categorization," *37th Chinese Control Conference (CCC),* pp. 9452-9455, 2018.

[11] T. Mikolov, K. Chen, G. Corrado and J. Dean, " Efficient estimation of word representations in vector space," 2013.

[12] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems ,* pp. 3111-3119, 2013.