Penerapan Weighted Word Embedding pada Pengklasifikasian Teks Berbasis Recurrent Neural Network untuk Layanan Pengaduan Perusahaan Transportasi

Muhammad David Rhman, Arif Djunaidy, dan Faizal Mahananto
Departemen Sistem Informasi, Institut Teknologi Sepuluh Nopember (ITS)
e-mail: adjunaidy@its.ac.id

Abstrak*—Twitter menjadi salah satu media sosial yang paling sering dan paling populer digunakan oleh perusahaan sebagai penyedia layanan pelanggan perusahaan. Adanya ribuan cuitan yang dapat masuk dalam setiap hari, tentu akan merepotkan operator layanan untuk mengkategorikan jenis berbagai cuitan tersebut, lebih-lebih jika proses pemilahan kategori cuitan harus dilakukan secara manual. Dalam Tugas Akhir ini, kategoriasi cuitan secara otomatis dibangun dan diimplementasikan menggunakan model klasifikasi berbasis recurrent neural network (RNN) yang dikombinasikan dengan model weighted word embedding (WWE). RNN merupakan salah satu jenis jaringan syaraf tiruan yang populer dan banyak digunakan dalam persoalan klasifikasi, sedangkan WWE merupakan metode yang memungkinkan untuk meng-hubungkan kata-kata yang serupa dengan mengukur jarak semantik antara vektor yang disematkan pada kata tersebut dan memberikan bobot yang berbeda pada setiap kata pada suatu kelas tertentu. Implementasi model penggabungan RNN dan WWE diuji coba menggunakan data pengaduan di perusahaan transportasi untuk data cuitan pada tahun 2015-2016. Hasil uji coba menunjukkan bahwa implementasi WWE baik yang menggunakan model FastText (Weighted FastText) maupun model Word2Vec (Weighted Word2Vec) memberikan hasil yang lebih baik dibandingkan dengan hasil kinerja yang menggabungkan RNN dan model word embedding biasa. Dengan menggunakan metode evaluasi berbasis 10-fold cross validation, model gabungan RNN-Weighted FastText dan RNN-Weighted Word2Vec berturut-turut memberikan hasil akurasi sebesar 88,2% dan 87,5%. Di lain pihak, dengan menggunakan metode evaluasi yang sama, model gabungan RNN-FastText dan RNN-Word2Vec memberikan hasil akurasi yang sama sebesar 83,4%.

***Kata Kunci*—FastText, Klasifikasi teks, Layanan Pengaduan Transportasi, Recurrent Neural Network, Twitter, Weighted Word Embedding, Word2Vec.**

I. PENDAHULUAN

PADA era digital ini, media sosial merupakan suatu platform penting yang ikut mendorong kemajuan teknologi khususnya dalam kehidupan sosial manusia. Media sosial saat ini tidak hanya digunakan sebagai ruang berekspresi dan beropini seperti sedia kala. Namun, saat ini sudah bertransformasi menjadi suatu platform yang digunakan untuk segala jenis interaksi sosial di masyarakat. Salah satu contohnya yaitu penggunaan media sosial sebagai tempat menjual produk berupa barang dan juga jasa. Selain itu juga menggunakan media sosial sebagai platform untuk mengiklankan produk yang dijual. Contoh lainnya yaitu penggunaan media sosial sebagai *customer service* (layanan pelanggan) dan support.

Saat ini ada banyak *platform* media sosial yang digunakan, beberapa contohnya yaitu Facebook, Twitter, Youtube, Instagram, dan lain-lain. Berdasarkan infografis dari GlobalWebIndex, pengguna aktif media sosial di Indonesia mencapai 130 juta orang. Twitter sendiri merupakan salah satu platform dengan pengguna yang lumayan besar jumlahnya. Menurut data yang diberikan Twitter, ada total 330 juta pengguna Twitter yang aktif setiap harinya, dengan total 126 juta cuitan yang masuk. Twitter menjadi salah satu platform yang paling sering digunakan oleh perusahaan sebagai penyedia layanan pelanggan mereka. Berdasarkan survei yang dilakukan Aberdeen Group, 41% dari 170 perusahaan yang mereka survei sudah menggunakan media sosial sebagai media untuk layanan pelanggan. Dari semua perusahaan yang memanfaatkan media sosial, 51% perusahaan menggunakan Twitter sebagai *platform* untuk layanan pelanggan mereka. Penggunaan Twitter sebagai customer services dan support secara daring banyak digunakan karena memeberikan informasi secara *real-time* tanpa harus menelpon layanan pengaduan. Pelanggan pun senang menggunakannya dikarenakan tidak perlu mengeluarkan biaya, cukup menggunakan internet.

Penggunaan media sosial sebagai sarana pengaduan pelanggan merupakan tantangan tersendiri untuk suatu perusahaan. Banyak pelanggan tentunya mengharapkan penganan aduan secara *real-time* seperti pada saat pelanggan melakukan aduan dengan menghubungi *call center*. Namun dengan media sosial hal ini akan cukup sulit karena banyaknya aduan yang masuk dalam satu waktu tanpa ada filterisasi. Waktu respons selalu menjadi nilai penting bagi pelanggan. Oleh karena itu pelanggan mengharapkan pengaduan menggunakan media sosial dapat ditanggapi secara *real-time,* setidaknya ditanggapi pada hari yag sama. Northridge Group melaporkan bahwa 42 persen konsumen mengharapkan tanggapan atas permintaan layanan pelanggan mereka dalam satu jam. Dari kelompok ini, 17 persen mengharapkan tanggapan dalam hitungan menit. Waktu respons memang penting namun ada hal lain yang tidak kalah peting yaitu menyelesaikan aduan yang dilaporkan [1].

Kategorisasi pada aduan yang masuk melalui Twitter dapat mengurangi masalah-masalah diatas. Filterisasi aduan yang masuk secara manual tentu akan memakan waktu yang sangat lama, terutama jika aduan masuk secara bersamaan dengan jumlah banyak. Namun dengan komputasi kategorisasi teks, hal ini dapat ditangani dengan cepat. Aduan yang masuk akan

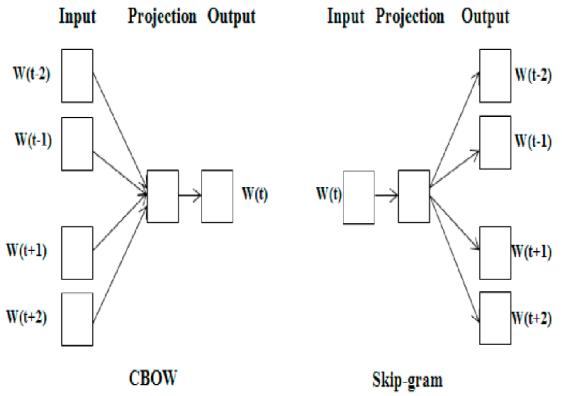
Input: a training set $\mathcal{C}$ of n classes.
Output: weighted word embeddings $\mathbf{I}_1, \ldots, \mathbf{I}_n$.
1: Use $\mathcal{C}$ to create a vocabulary $\mathcal{V}$ of words.
2: Cluster the documents in $\mathcal{C}$ with the same label a $\{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$.
3: **for** each $i \in \{1, \ldots, n\}$ **do**
4: Predefine a weight vector $\mathbf{w}_i \in \mathbb{R}^{|\mathcal{V}|}$;
5: Obtain a TF-IDF term-document matrix $\mathcal{M}_i$ with $\mathcal{C}_i$;
6: For the words occurring in $\mathcal{C}_i$, assign the maximur value of each column of $\mathcal{M}_i$ to corresponding entr of $\mathbf{w}_i$;
7: For the words not occurring in $\mathcal{C}_i$, assign the mini mum value of all maximum values to the rest entrie of $\mathbf{w}_i$;
8: **end for**
9: For a sentence $\mathbf{s}$ in $\mathcal{C}$, pad or truncate it to length k;
10: Get the word embedding matrix $\mathbf{E} \in \mathbb{R}^{k \times l}$ of $\mathbf{s}$;
11: **for** each $i \in \{1, \ldots, n\}$ **do**
12: **for** each $j \in \{1, \ldots, k\}$ **do**
13: For the jth word in $\mathbf{s}$, find its corresponding weigh $\hat{w}_{ij}$ from $\mathbf{w}_i$;
14: **end for**
15: Denote $\hat{\mathbf{w}}_i = (\hat{w}_{i1}, \ldots, \hat{w}_{ik})^{\mathrm{T}}$, i.e., a weight vector com puted from the collection $\mathcal{C}_i$;
16: Let $\mathbf{W}_i$ be a $k \times l$ matrix whose each column is $\hat{\mathbf{w}}_i$;
17: Let $\mathbf{I}_i = \mathbf{W}_i \odot \mathbf{E}$, where $\odot$ denotes the element-wis multiplication of two matrices.
18: **end for**

Gambar 3. Pseudocode WWE.

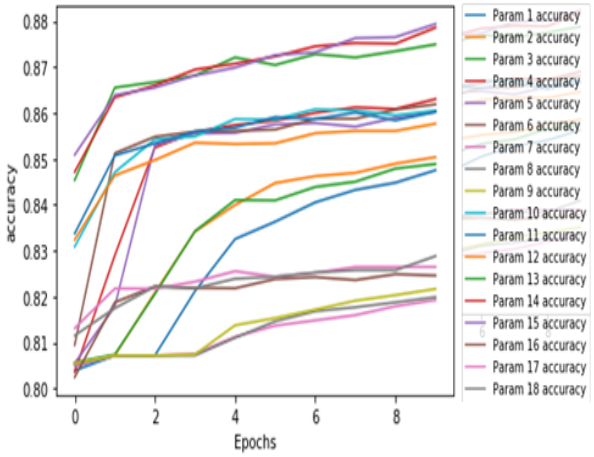
terpilah-pilah. Dan ini akan memudahkan dalam penyelesaian aduan tersebut. Sehingga tidak hanya memberikan efek yang baik pada waktu respons aduan, namun juga perusahaan dapat menyelesaikan masalah yang diadukan oleh pelanggan.

Kategorisasi cuitan dapat dilakukan dengan berbagai cara. Saat ini sudah banyak tools kategorisasi Twitter yang dapat digunakan untuk mengelompokkan cuitan pada Twitter. Cara lain yang dapat digunakan adalah dengan melakukan klasifikasi teks. Klasifikasi teks ini dapat digunakan menggunakan ilmu text mining (penggalian teks). Klasifikasi teks merupakan sebuah proses pembelajaran mesin yang mampu mengkategorikan dokumen teks ke dalam kelas-kelas yang sudah ditentukan sebelumnya secara tepat. Sehingga cuitan yang masuk akan dapat langsung dikategorikan ke dalam kelas yang sudah ditentukan secara otomatis. Dalam melakukan klasifikasi teks dapat dilakukan dengan berbagai metode, yaitu *Random Forest, Naive Bayes, Support Vector Machine* (SVM), dan *Neural Network*. *Neural Network* atau metode jaringan syaraf merupakan metode pembelajaran mesin yang berdasar pada pembelajaran yang menyerupai sistem otak manusia dalam menangkap pola. Metode ini sedang digandrungi dalam melakukan kalsifikasi teks dibandingkan metode konvensional lainnya. Ide penggunaan metode ini adalah untuk menyelesaikan permasalahan *sparsity* data dan representasi semantik dari data [2].

Model *neural network* sudah banyak digunakan untuk melakukan representasi kata, model representasi ini biasanya disebut *word embedding*. *Word embedding* merupakan suatu metode penyisipan kata, mengubah kata menjadi suatu vektor yang kontinyu dengan panjang yang sudah ditetapkan, sehingga tidak akan dibatasi dengan kosakata yang lebih banyak. Model ini juga kaya akan informasi semantik, sehingga memungkinkan kita untuk menghubungkan katakata yang serupa dengan mengukur jarak semantik antara vektor yang disematkan pada kata tersebut. Namun pada



Gambar 1. Arsitektur Word2Vec.



Gambar 2. Hasil perbandingan antar skenario.

pengaplikasiannya, *word embedding* biasa tidak menaruh bobot pada tiap kata pada suatu kelas. Sedangkan, pada umumnya tiap kata akan memiliki makna dan kepentingan yang berbeda pada penggunaannya. Oleh karena itu, pada tugas akhir ini diusulkan menggunakan metode *weighted word embedding*. Dimana tiap kata akan memiliki bobot yang berbeda tergantung pada kelas mana kata tersebut dikategorikan.

Neural network memiliki banyak metode yang dapat digunakan untuk melakukan klasifikasi teks. Salah satu yang populer dan seing digunakan adalah metode *Recurrrent Neural Network*. Metode ini memiliki karakteristik tersendiri yaitu memiliki "memori" yang dapat menyimpan prediksi kata sebelumnya sebagai masukkan prediksi kata selanjutnya. Hal ini dapat meningkatkan akurasi dalam melakukan prediksi berkelanjutan. Umumnya metode jaringan syaraf memiliki input dan output yang independen, tidak bergantung satu sama lain, namun beberapa kasus butuh kesinambungan antara satu kalimat dengan kalimat lain yang akan diprediksi. Dengan menggunakan *hidden layer* RNN masalah tersebut dapat diselesaikan. RNN mampu menangkap konteks secara luas dari data karena karakteristiknya yang memiliki dependensi antar kata jangka panjang [3].

Pada tugas akhir ini akan membuat sebuah model klasifikasi teks dengan menggunakan metode *recurrent neural network* yang akan dipadukan dengan *weighted word embedding* untuk menghasilkan model yang akurat. Model klasifikasi nantinya akan di aplikasikan pada data Twitter layanan pengaduan perusahaan trasportasi pada kasus ini

Tabel 1.
Jumlah label kategori setelah praproses

Kategori	Jumlah
Informasi	15.746
Keluhan	3.141
Saran	562

Tabel 2.
Perbandingan Skenario

	Model *Word Embedding*	**Parameter**	**Panjang Vektor**	**Fungsi Aktivasi**
	WWE-W2V	Parameter 1	50	Aktivasi *softmax* pada
Skenario 1		Parameter 2	100	tiap layer
		Parameter 3	200	
	WWE-FT	Parameter 4	50	
Skenario 2		Parameter 5	100	
		Parameter 6	200	
	Word2Vec	Parameter 7	50	
Skenario 3		Parameter 8	100	
		Parameter 9	200	
	WWE-W2V	Parameter 10	50	Aktivasi *relu* pada tiap
Skenario 4		Parameter 11	100	layar dan aktivasi
		Parameter 12	200	*softmax* pada *output*
	WWE-FT	Parameter 13	50	*layer*
Skenario 5		Parameter 14	100	
		Parameter 15	200	
	Word2Vec	Parameter 16	50	
Skenario 6		Parameter 17	100	
		Parameter 18	200	

yaitu PT. Kereta Api Indonesia (Persero), yaitu pada akun @KAI121. Model ini juga akan dibandingkan menggunakan data lainnya sebagai bahan validasi keakuratan model.

II. METODE PENELITIAN

A. *Recurrent Neural Network*

Recurrent neural network merupakan metode pembelajaran mesin yang berdasar pada pembelajaran jaringan syaraf (*neural network*) yang menyerupai sistem otak manusia dalam menangkap pola. RNN adalah satu dari sekian banyak metode jaringan syaraf yang dapat digunakan. Berbeda dengan metode jaringan syaraf lainnya RNN terkenal dengan kemampuannya melatih prediksi berdasarkan prediksi sebelumnya [2]. Hal ini dikarenakan RNN memiliki ”memori” yang menyimpan informasi dari hasil kalkulasi sebelumnya. Umumnya metode jaringan syaraf memiliki *input* dan *output* yang independen, tidak bergantung satu sama lain, namun beberapa kasus butuh kesinambungan antara satu kalimat dengan kalimat lain yang akan diprediksi. Dengan menggunakan *hidden layer* RNN masalah tersebut dapat diselesaikan. RNN mampu menangkap konteks secara luas dari data karena karakteristiknya yang memiliki dependensi antar kata jangka panjang [3].

Setiap RNN memiliki perulangan, apabila dalam 1 kalimat terdapat 3 kata maka akan terdapat 3 sekuens *layer neural network* dengan masing-masing 1 *layer* untuk tiap kata, seperti gambar 1 saat skema tersebut dibuka (*unfold*) [3]. Pada setiap pembelajaran RNN akan dimulai dengan *input* (xt). Lalu selanjutnya dilakukan perhitungan untuk *hidden state* (s) dengan menggunakan *parameter* (U dan W). Dari perhitungan *hidden state* maka dapat digunakan untuk menghitung *ouput* (ot). Berikut rumus-rumus yang digunakan untuk perhitungan:

$$Current\ State\ (s_t) = f(Ux_t + Ws_{t-1}) \quad (1)$$

$$Ouput\ (o_t) = V.s_t \quad (2)$$

Parameter U,V,dan W diatas memiliki nilai yang sama pada tiap *layer*-nya. Hal inilah yang membedakan RNN dengan *deep neural network* lainnya. Fungsi f pada rumus perhitungan Current State (1) dapat menggunakan *tanh* atau *ReLu.*

B. *Weghted Word Embedding*

Word embedding merupakan suatu metode penyisipan kata, mengubah kata menjadi suatu vektor yang kontinyu dengan panjang yang sudah ditetapkan, sehingga tidak akan dibatasi dengan kosakata yang lebih banyak. Pembobotan kata merupakan suatu strategi praproses data dengan menetapkan bobot yang sesuai untuk setiap istilah sehingga bobot mewakili relevansi istilah dengan dokumen, model ini memainkan peran penting dalam meningkatkan kinerja klasifikasi teks. Model pembobotan tradisional, seperti *Term Frquency* (TF) dan *Term Frequency Inverse Document Frequency* (TF-IDF), bersifat *unsupervised* [3], yang berarti model tersebut tidak menggunakan informasi label dokumen untuk menghitung bobot. Semua metode pembobotan yang biasa digunakan mengikuti aturan yang sama, yaitu, satu istilah hanya memiliki satu bobot, yang secara implisit mengasumsikan bahwa istilah yang sama juga memiliki makna yang sama dalam dokumen yang berbeda. Namun, sangat umum bahwa istilah muncul dalam dokumen dengan label berbeda dan dalam tiap dokumen, masing-masing kata dapat memiliki makna (kepentingan) yang berbeda. Untuk mengatasi kekurangan tersebut, digunakanlah strategi pembobotan baru untuk klasifikasi teks dimana setiap istilah akan diberikan bobot yang berbeda tergantung pada kelasnya. Model TF-IDF tetap digunakan untuk melakukan pembobotan ini karena modelnya sederhana dan mempunyai kinerja yang baik [4]. TF-IDF merupakan suatu pengukuran statistik yang digunakan secara luas untuk mengevaluasi

Tabel 3.
Hasil 10-Fold Cross Validation

Fold	*Accuracy*	*Loss*
1	85.15%	0.4025
2	84.51%	0.3889
3	86.82%	0.3643
4	85.79%	0.3875
5	83.68%	0.4285
6	85.66%	0.3781
7	85.28%	0.3911
8	86.05%	0.3837
9	86.17%	0.3573
10	86.10%	0.3955

Tabel 4.
Hasil prediksi pada data teks Twitter

Teks	Label	Prediksi
siang kai bisa intern kuliah semester fakultas manajemen	Informasi	Informasi
berniat beli tiket kursi penumpang kursi kereta eko progo	Informasi	Informasi
penasaran penumpang telat tiket hangus keretanya telat berangkat kompensasinya eks lho	Keluhan	Keluhan
ditemukan salah tempel huruf kursi ka harina kertajaya	Informasi	Informasi

seberapa pentingkah sebuah kata bagi sebuah dokumen dalam sebuah korpus [3]. Contoh, diberikan kosakata istilah $\mathfrak{V}$ dan kumpulan dokumen $\mathfrak{D}$, bobot TF-IDF untuk suatu istilah $t \in \mathfrak{V}$ dan dokumen $d \in \mathfrak{D}$ dikomputasikan dalam perhitugan berikut:

$$TF(t,d) = f_{t,d},$$

$$IDF(t,d) = \log\left(\frac{|\mathfrak{D}|}{m}\right) + 1, \quad (3)$$

$$TF\text{-}IDF(t,d) = TF(t,d) \times IDF(t,d)$$

Dimana, $f_{t,d}$ = Frekuensi munculnya istilah t pada dokumen d, $|\mathfrak{D}|$ merepresentasikan jumlah dokumen sedangkan m adalah jumlah dokumen pada $\mathfrak{D}$ yang memiliki istilah t.

Pada gambar 1 menunjukkan *pseudocode* dari model *weighted word embedding.*Proses *weighted word embedding* dapat dibagi menjadi dua bagian proses, yaitu komputasi pemberian bobot ke masing-masing kata dan komputasi pembuatan *weighted word embedding. Weighted word embedding* dapat dihasilkan dari perkalian antara matriks ***W***, hasil atau vektor dari pembobotan tiap kata dalam satu dokumen, dan matriks ***E***, matriks transpose yang berisikan word embedding yang didapat secara acak ataupun menggunakan pre-trained vectors yang dibuat oleh Word2Vec dan FastText.

1) Word2Vec

Word embedding merupakan suatu permodelan bahasa dan fitur teknik pembelajaran di *Natural Language Processing* di mana kata atau frasa diwakili dalam bentuk vektor bilangan real. Konsep *word embedding* melibatkan rumus matematika. Model yang digunakan dalam word embedding sangat beragam, salah satunya adalah model Word2Vec. Word2Vec merepresentasikan kata-kata menjadi vektor berdasarkan beberapa fitur yang dimilikinya seperti ukuran dan dimensi vektor.

Tabel 5.
Confussion Matrix hasil prediksi label kategori

Data Aktual	Prediksi		
	Informasi	Keluhan	Saran
Informasi	2984	161	1
Keluhan	230	387	2
Saran	61	57	6

Tabel 6.
Performa presisi, akurasi, recall, dan f-measure dari model

Precision	0.860
Recall	0.868
Accuracy	0.868
F-measure	0.855

Model Word2Vec diusulkan oleh Mikolov dengan keunggulan representasi vektor ini mampu menangkap sintaks dan makna semantik dari kata menggunakan *NLP (Natural Language Programming)*. Selain itu, model Word2Vec adalah algoritma representasi vektor kata yang mampu mencapai kinerja terbaik di *NLP* dengan mengelompokan kata yang sama, yaitu kata-kata yang sama memiliki vektor yang sama atau vektor berdekatan. Nilai kesamaan dari vektor memiliki rentang dari -1 hingga 1 (tertinggi).

Arsitektur model Word2Vec dihitung dengan menggunakan *Neural Network* dengan kata teks sebagai input dan ruang vektor sebagai outputnya. Vektor kata yang dihasilkan adalah vektor ruang dimensi yang menangkap makna semantik kata tersebut. Ada dua jenis model arsitektur Word2Vec, yaitu model *Skip-Gram* dan *CBOW (Continuous Bag of Words)*. Model *Skip-gram* diperkenalkan sebagai metode yang efisien untuk mempelajari vektor kata dalam teks yang tidak terstruktur dalam jumlah yang besar. Arsitektur model *Skip-gram* membuat prediksi kata dengan hanya menggunakan sebagian kata atau bahkan satu kata dalam suatu teks yang tidak terstruktur, sedangkan model CBOW memprediksi kata berdasarkan kata konteks kata secara keseluruhan.

2) FastText

FastText merupakan suatu model *word embedding* yang dikembangkan oleh Facebook. FastText ini adalah pengembarngan dari model *word embedding* Word2Vec. FastText mampu mencapai kinerja yang sangat baik untuk representasi kata dan klasifikasi kalimat, khususnya dalam kasus kata-kata langka dengan memanfaatkan informasi tingkat karakter.

Pada model FastText setiap kata direpresentasikan dengan kumpulan karakter dengan panjang *n-gram*. Semisal kata ‘representasi’, dengan nilai *n-gram* = 3 maka pada model FastText akan diproses mejadi (‘rep’, ‘epr’, ‘pre’, ‘res’, ‘ese’,’sen’, ‘ent’, ‘nta’, ‘asi’). Dengan merepresentasikan kata menjadi karakter sejimlah *n-gram*, model ini mampu mengklasifikasikan kata yang langka dan juga mendeteksi apabila terdapat kesalahan penulisan dalam kata.

III. HASIL DAN DISKUSI

A. Lingkungan Uji Coba

Lingkungan uji coba yang digunakan untuk proses uji coba model adalah laptop dengan *processor* Intel i7-6500U dan

memori 8192 MB. Sistem operasi menggunakan Windows 10, dengan Python dan Spyder sebagi *Integrated Development Enviroment* (IDE).

B. Hasil Praproses Teks

Data pengaduan yang masuk pada akun media sosial Twitter ini memiliki panjang 20.190 baris. Setelah melalui fase praproses pada pembahasan sebelumnya, yaitu Twitter *text cleaning*, menghapus data *null*, menghapus *stopword*, *stemming* menggunakan SastrawiStemmer [5], dan proses pembersihan data lainnya. Data bersih yang sudah di praproses memiliki panjang 19.449 baris. Data memiliki tiga buah label kategori yaitu informasi, keluhan dan saran/permintaan. Jumlah tiap label kategori dapat dilihat pada tabel 1.

Setelah semua data selesai di praproses dan sudah bersih siap untuk di proses. Hal yang harus dilakukan pertama adalah membagi data menjadi dua, yaitu data *training* (latih) dan data *test* (uji). Perbandingan data latih dan uji adalah 80:20. Data latih mengambil 80% dari data awal, sedangkan data uji mengambil sisanya yaitu 20%. Jumlah data latih terdapat 15.559 baris sedangkan data uji sebanyak 3.890 baris.

C. Model Word Embedding

Pada *word embedding*, seperti yang ssudah dijelaskan pada pembahasan sebelumnya menggunakan *weighted word embedding*. *Weighted Word Embedding*. Data latih dan uji yang sudah bersih dilakukan *embedding* menggunakan model *weighted word embedding*. Hal ini bertujuan agar model dapat berjalan dengan efektif. Model ini merupakan model word embedding berbasis Word2Vec dan FastText. Dan juga TF-IDF *Library* yang digunakan dalam membuat model ini yaitu TfidfVectorizer dan gensim.

D. Skenario Model

Pada pengerjaan tugas akhir ini akan membandingkan beberapa skenario model untuk mencari mana hasil yang terbaik. Pada pembuatan skenario ini ada beberapa parameter yang akan diubah-ubah, yaitu *size* dan *length* dari vektor, *activation* pada masing-masing *layer* RNN, dan yang terakhir yaitu model *word embedding*.

Terdapat tiga model *word embedding* yang akan dicoba pada tugas akhir ini yaitu *Weighted Word Embedding* menggunakan Word2Vec (WWE-W2V), *Weighted Word Embedding* menggunakan FastText (WWE-FT), dan model *word embedding* menggunakan Word2Vec saja. Sedangkan untuk *size* dan *length* dari vektor hasil *word embedding* akan dibandingkan tiga hasil yaitu dengan panjang 50, 100, dan 200. Selain skenario dari *word embedding*, penulis juga membandingkan parameter *activation* yang ada pada tiap *layer* RNN. Arsitektur RNN yang digunakan adalah arsitektur LSTM (Long-Short Term Memory). Pada skenario fungsi aktivasi ini akan mencoba dua jenis fungsi aktivasi yaitu *relu* dan *softmax*. Pada masing-masing *layer* akan dilakukan skenario pemberian fungsi aktivasi *relu* dan aktivasi *softmax*. Detail skenario dapat dilihat pada tabel 2.

Terdapat enam skenario yang nantinya akan dibandingkan hasilnya. Sebelum membandingkan antar skenario, hal pertama yang dilakukan adalah membandingkan masing-masing percobaan yang terdapat di dalam skenario. Lalu membandingkan mana yang terbaik antara skenario 1 hingga skenario 6.

E. Perbandingan

Dari 6 skenario dan 18 parameter yang dibandingkan. Terdapat 1 skenario yang memiliki skor akurasi yang paling tinggi yaitu skenario 5. Pada skenario 5 dengan panjang vektor 200 memiliki akurasi hingga 0,88. Berikut grafik perbandingan tiap parameter yang dapat dilihat pada gambar 3.

Dari gambar 2 dapat dilihat kalau skenario 5 memiliki akurasi yang paling unggul diantara semua skenario. Parameter 18 mengahasilkan akurasi tertinggi yaitu 0,88. Penggunaan FastText yang diberikan bobot memang memberikan dampak pada performa model klasifikasi. Vektor dengan panjang 200 memang tidak memberikan kenaikan akurasi yang signifikan dibandingkan dengan vektor dengan panjang 50 dan 100. Namun memberikan hasil yang sedikit lebih konsisten dan memberikan hasil akurasi yang paling tinggi dibandingkan yang lainnya. Selanjutnya Skenario 5 dengan parameter 18 akan dijadikan model yang dipilih untuk melakukan prediksi namun sebelum itu akan dilakukan validasi terlebih dahulu.

F. Validasi Model

Untuk memvalidasi model klasifikasi yang dibuat pada tugas akhir ini penulis menggunakan *K-fold Cross Validation* dengan nilai k sebesar 10. Validasi model dilakukan menggunakan data validasi yang sudah diambil 10% dari data pelatihan. Hasil dari validasi model ini dapat dilihat dari tabel 3. Rerata dari hasil akurasi dan *loss* secara berturut-turut adalah 85,54% dan 0,3895.

G. Prediksi Model

Setelah mendapatkan model yang paling bagus dari mencoba setiap skenario dan melakukan validasi pada model tersebut. Hal selanjutnya yang dilakukan yaitu melakukan uji prediksi pada model menggunakan data uji (*test*). Sebelum melakukan prediksi data kita melakukan evaluasi model terlebih dahulu untuk melihat hasil akurasi dan *loss* pada model. Hasil akurasi yang didapatkan pada data pengujian adalah 0,8683. Akurasi ini lebih kecil dibanding pada data pelatihan. Skor *loss* yang dihasilkan dari evaluasi ini dapat dibilang kecil dan ini menandakan bahwa model tidak *overfitting*.Selanjutnya kita langsung melakukan prediksi pada model yang dapat dilihat pada tabel 4.

Dari hasil prediksi model ingin dilihat bagaimana performa model dan seberapa tepat serta akurat model dapat memprediksi label dari tiap data teks. Berikut *confussion matrix* dari hasil prediksi menggunakan data uji yang dapat dilihat pada tabel 5.

Dari 3889 data pengujian, terdapat 3146 berkategori Informasi, 619 berkategori Keluhan dan 124 berkategori Saran/Permintaan. Model RNN dengan metode *word embedding* menggunakankan *weighted* FastText ini memberikan hasil akurasi sebesar 0,868. Model ini dapat dengan secara tepat memprediksi 2984 kategori Informasi, 387 kategori Keluhan, dan 6 kategori Saran/Permintaan.

Pada kategori Saran/Permintaan hanya sedikit teks yang secara tepat diprediksi. Hal ini menurut penulis dikarenakan oleh sedikitnya data pelatihan pada kategori tersebut. Selain

itu adanya ambiguitas kata pada kategori tersebut yang juga terdapat pada kategori Informasi dan kategori Keluhan yang notabene memiliki lebih banyak data pelatihan. Secara kategori, model dapat mempediksi kategori Informasi secara tepat dengan skor presisi 0,911 sedangakn untuk kategori Keluhan dan Saran/Permintaan secara bertutrut-turut adalah 0,640 dan 0,667.

Hasil performa data pengujian secara keseluruhan dapat dilihat pada tabel 6. Hasil *precison*, *recall*, *accuracy*, dan *f-measure* dari model RNN dengan *Weighted Word Embedding* FastText secara berturut-turut adalah 0,860; 0,868; 0,868; dan 0,855.

IV. KESIMPULAN/RINGKASAN

Pada tugas akhir ini ingin membuktikan bahwa penggunaan *word embedding* dengan model *weighted embedding* memiliki dampak yang cukup signifikan dalam menambah akurasi model dibandingkan dengan menggunakan *layer embedding* biasa. Dengan menggunakan model *weighted word embedding* menggunakan FastText memiliki akurasi paling tinggi dengan nilai 88,2% dengan fold sejumlah 10. Sedangkan *weighted word embedding* menggunakan Word2Vec menghasilkan akurasi sebesar 87,5%. Hal ini jelas lebih baik dibandingkan hanya melakukan *embedding* dengan menggunakan model Word2Vec biasa karena hanya menghasilkan akurasi sebesar 83,4%.

DAFTAR PUSTAKA

[1] M. Consulting and Q. Monitoring, “State of Customer Service Experience The Northridge Group ’ s.” The Northridge Group, Inc, West Higgins Road, Suite, pp. 1–15, 2018.

[2] Hoon-KengPoon, W.-S. Yap, Y.-K. Tee, W.-K. Lee, and B.-M. Goi, “Hierarchical gated recurrent neural network with adversarial and virtual adversarial training on text classification,” *Neural Networks*, vol. 119, pp. 299–312, 2019.

[3] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *IEEE Trans. Neural Networks*, vol. 5, no. 2, pp. 157–166, 1994, doi: 10.1109/72.279181.

[4] P. Liu, X. Qiu, and H. Xuanjing, “Recurrent Neural Network for Text Classification with Multi-Task Learning,” in *IJCAI International Joint Conference on Artificial Intelligence*, 2016, pp. 2873–2879.

[5] B. Guo, C. Zhang, J. Liu, and X. Ma, “Improving text classification with weighted word embeddings via a multi-channel TextCNN model,” *Neurocomputing*, vol. 363, pp. 366–374, 2019, doi: 10.1016/j.neucom.2019.07.052.