Sentiment Analysis On YouTube Comments Using Word2Vec and Random Forest

Sentimen Analisis pada Opini YouTube Menggunakan Word2Vec dan Random Forest

Siti Khomsah

Program Studi Sains Data, Institut Teknologi Telkom Purwokerto, Jawa Tengah, Indonesia

siti@ittelkom-pwt.ac.id

Informasi Artikel
Received: 21 December 2020
Revised: 15 January 2021
Accepted: 23 February 2021
Published: 28 February 2021

Abstract

Purpose*: This study aims to determine the accuracy of sentiment classification using the Random-Forest, and Word2Vec Skip-gram used for features extraction. Word2Vec is one of the effective methods that represent aspects of word meaning and, it helps to improve sentiment classification accuracy.*
Methodology*: The research data consists of 31947 comments downloaded from the YouTube channel for the 2019 presidential election debate. The dataset consists of 23612 positive comments and 8335 negative comments. To avoid bias, we balance the amount of positive and negative data using oversampling. We use Skip-gram to extract features word. The Skip-gram will produce several features around the word the context (input word). Each of these features contains a weight. The feature weight of each comment is calculated by an average-based approach. Random Forest is used to building a sentiment classification model. Experiments were carried out several times with different epoch and window parameters. The performance of each model experiment was measured by cross-validation.*
Result*: Experiments using epochs 1, 5, and 20 and window sizes of 3, 5, and 10, obtain the average accuracy of the model is 90.1% to 91%. However, the results of testing reach an accuracy between 88.77% and 89.05%. But accuracy of the model little bit lower than the accuracy model also was not significant. In the next experiment, it recommended using the number of epochs and the window size greater than twenty epochs and ten windows, so that accuracy increasing significantly.*
Value*: The number of epoch and window sizes on the Skip-Gram affect accuracy. More and more epoch and window sizes affect increasing the accuracy.*

Keywords:youtube-comments, sentiment analysis, word2vec,skip-gram, random forest.
Kata kunci: komentar youtube, analisis sentimen, word2vec, skip-gram, random forest.

Abstrak

Tujuan: Penelitian bertujuan mengetahui akurasi klasifikasi sentimen menggunakan Random Forest dengan ekatraksi fitur Word2Vec model Skip-gram. Word2Vec mencari kedekatan makna semantik sebuah kata, sehingga metode ini dianggap lebih baik dari pendekatan leksikal.

Metode: Data penelitian sebanyak 31947 komentar yang diunduh dari kanal YouTube debat pemilu presiden 2019. Dataset terdiri 23612 komentar positif dan 8335 komentar negatif. Untuk menghindari bias, penyeimbangan jumlah data positif dan negatif menggunakan *oversampling*. Ekstraksi fitur menggunakan Word2Vec jenis Skip-gram. Metode Skip-gram akan menghasilkan *output* beberapa fitur disekitar kata yang dijadikan *input*. Setiap fitur tersebut mengandung bobot. Bobot fitur setiap komentar dihitung dengan pendekatan *average based*. Random Forest digunakan untuk membangun model klasifikasi sentimen. Percobaan dilakukan beberapa kali dengan parameter *epoch* dan *window* yang berbeda-beda. Performa setiap percobaan model diukur dengan *cross- validation*.

Hasil: Berdasarkan percobaan dengan parameter 1, 5, dan 20 *epoch* dan ukuran *window* 3, 5, dan 10, didapat rata-rata akurasi model antara 90,1 % sampai 91%. Tetapi pengujian model menghasilkan akurasi antara 88,77% sampai dengan 89,05%. Akurasi model saat diuji justru turun meskipun tidak signifikan. Percobaan dengan jumlah epoch dan ukuran *window* sebaiknya menggunakan nilai lebih besar dari 20 *epoch* dan 10 *window* sehingga bisa dilihat peningkatan akurasi yang signifikan.

State of the art: *Epoch* dan ukuran *window* pada Skip-Gram mempengaruhi akurasi, semakin tinggi nilai *epoch* dan ukuran *window* berpengaruh pada peningkatan akurasi model.

1. Pendahuluan

Analisis sentimen YouTube bertujuan mengekstraksi polaritas opini dari dataset komentar pengguna. Komentar terhadap video YouTube menunjukkan sentimen positif maupun negatif yang diwakili oleh rasa senang sampai dengan sedih atau tidak suka. Secara umum, tahapan model analisis sentimen dari data komentar meliputi pembersihan (*text-cleansing*), ekstraksi fitur kata, dan klasifikasi sentimen. Ekstraksi fitur kata adalah tahapan penting karena proses tersebut menentukan akurasi model klasifikasi pada tahap berikutnya.

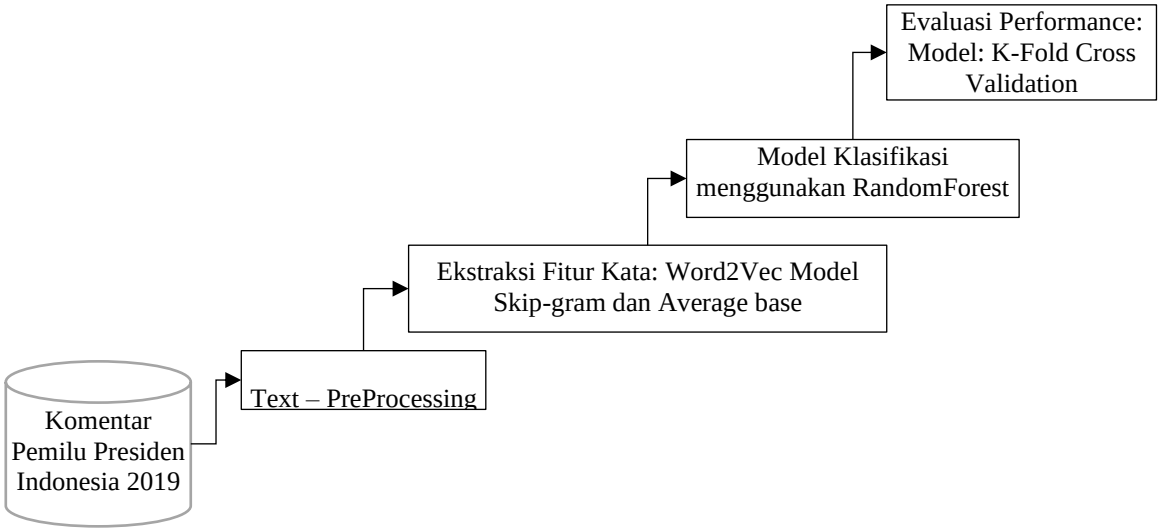
Term frequency (TF) dan Term-frequency-invers-term-frequency (TF-IDF) adalah metode statistik yang sering digunakan sebagai vektorisasi kata dalam *text mining*. TF

merepresentasikan kata berdasarkan tingkat kemunculan dalam dokumen. Sedangkan TF-IDF merepresentasikan distribusi frekuensi kata terhadap keseluruhan dokumen, diharapkan fitur kata yang dihasilkan semakin unik. Fitur yang unik ini dianggap dapat mewakili kata yang paling penting dalam korpus. Secara umum, TF-IDF lebih baik dibandingkan TF dalam meningkatkan akurasi model klasifikasi [1], dan lebih baik dibandingkan metode N-Gram[2]. Meskipun TF-IDF menghasilkan fitur unik namun fitur tunggal yang dihasilkan dapat bermakna samar. Integrasi metode gabungan kata (kolokasi) berdasarkan *part of speech* (POS) untuk memperbaiki cara kerja TF-IDF sehingga meningkatkan akurasi model pengklasifikasi[3]. TF-IDF akan menghasilkan ukuran matrik fitur kata yang sangat besar. Selain itu, fitur kata yang dihasilkan tidak mempertimbangkan keterhubungan dengan kata-kata lainnya secara semantik dalam dokumen (komentar) yang sama maupun dalam satu korpus[4]. Setiap kata mempunyai makna berbeda jika dipandang keterhubungannya dengan kata-kata lainnya dalam satu dokumen atau korpus tersebut. Contoh komentar, 'capres terburuk', memiliki makna berbeda jika dianggap sebagai dua term terpisah 'capres' dan 'terburuk'.

Word embedding adalah model pembelajaran yang menghasilkan representasi kata yang terdistribusi kontinu dalam ruang dimensi rendah. Secara umum. model pembelajaran yang digunakan adalah jaringan saraf tiruan (JST)[5]. Salah satu metode *word embedding* terkenal adalah Word2Vec yang ditemukan oleh Tomas Mikolov [6]. Word2Vec merupakan terobosan ektraksi fitur kata karena mencari semantik fitur kata dari korpus. Word2Vec mewakili setiap kata unik dengan sejumlah angka-angka yang disebut vektor. Vektor yang dipilih menggunakan fungsi matematis menunjukkan tingkat kemiripan secara semantik antara kata-kata yang diwakili vektor tersebut. Setiap kata memiliki ukuran vektor yang ditentukan oleh programmer, dimensi vektor bisa sampai 100[7]. Hal ini berbeda dengan TF maupun TF-IDF dimana setiap kata hanya direpresentasikan dalam satu nilai vektor saja. Akurasi model sentimen *product review* menggunakan SVM dan Word2Vec lebih rendah daripada TF-IDF, hal ini karena Word2Vec kurang dapat belajar dari dataset yang hanya 772 [8]. Word2Vec efektif jika diterapkan pada korpus yang besar, seperti kamus bahasa[9]. *Window* adalah salah satu parameter Word2Vec yang mempengaruhi akurasi model[10]. Penelitian ini bertujuan menerapkan Word2Vec untuk mengesktraksi fitur kata dari dataset komentar YouTube, dan melihat efeknya akurasi model klasifikasi. Metode klasifikasi yang digunakan adalah *ensemble machine learning* dan Random Forest.

2. Metode/Perancangan

Perancangan penelitian analisis sentimen yang dibangun ditunjukkan oleh Gambar 1.



Gambar 1 Perancangan Penelitian

2.1. Dataset

Dataset untuk penelitian merupakan data sekunder, diambil penelitian sebelumnya [11]. Dataset tersebut merupakan komentar- komentar di kanal YouTube yang menayangkan debat calon presiden Indonesia pada 2019. Sumber data dijelaskan pada Tabel 1.

Tabel 1. Rincian Dataset

Debat Ke-	Tema	Kanal	Jumlah komentar		
			Positif	**Negatif**	**Total**
I	Law, Human Rights, Corruption and Terrorism	Kompas TV IDN Times	4307	1744	6051
II	Energy and Food, Natural Resources and the Environment, and Infrastructure	MNC TV CNN Indonesia	1484	407	1891
III	Education, Health, Employment, Social and Culture	Kompas TV IDN Times	4066	1068	5134
IV	Ideology, Governance, Defense and Security, International Relations	Kompas TV IDN Times	1976	645	2621
V	Economy and Social Welfare, Finance and Investment, Trade and Industry	TV One News CNN Indonesia	11779	4471	16250
	Grand Total		**23612**	**8335**	**31947**

2.2. Text- Preprocessing

Preprocessing meliputi proses penghapusan tanda baca atau simbol lainnya seperti *hastag* (#), @, angka, dan simbol-simbol lain yang bukan merupakan kata. Penghapusan kata-kata yang

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI: 10.31515/telematika.v18i1.4493

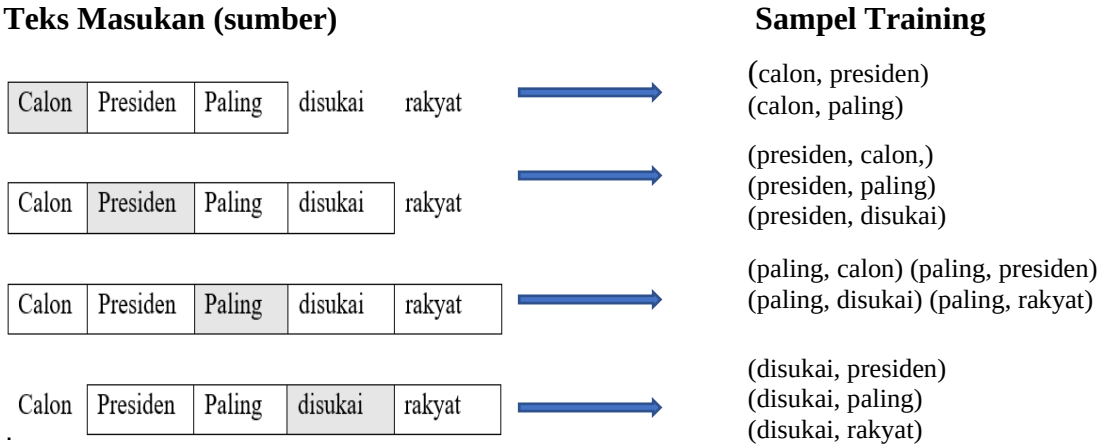
tidak mempunyai makna (*stopword removing*) menggunakan fungsi yang dikembangkan dari penelitian sebelumnya[12], dengan menambahkan konversi slang word[13].

2.3. Ekstraksi Fitur Kata dengan Word2Vec

Word2Vec menerapkan cara kerja jaringan saraf tiruan (JST) untuk merepresentasikan fitur kata yang mempunyai tingkat kemiripan semantik yang tinggi. Ada dua arsitektur yang digunakan yaitu *continuous bag-of-words* (CBOW) dan Skip-gram [7]. Penelitian ini menggunakan model Skip-gram, cara kerjanya memprediksi kata-kata disekitar (*target word)* kata yang diberikan (*context word*). Dalam menentukan target word, digunakan ukuran *window*, yang menunjukkan jumlah kata yang diamati pada posisi sesudah dan sebelum *context word*[6][14]. Ilustrasi Gambar 2 menunjukkan Skip-gram dengan ukuran *window* 2.

Contoh opini: calon presiden paling disukai rakyat

Jika diberikan *contex word* 'calon' maka target word-nya sebelah kiri kosong sedangkan sebelah kanan adalah 'presiden', dan ' paling'. Dengan demikian pasangan {*contex word, target word*} yang dihasilkan adalah {(calon, presiden), (calon, paling)}.



Gambar 2. Ilustrasi *Window* Model Skip-Gram

Pasangan {*contex word, target word*} inilah yang akan menjadi *input* dan *output* pembelajaran mesin, dalam arsitektur Skip-gram. Cara kerja ekstraksi fitur terdiri langkah-langkah berikut[15][16].

Langkah 1. Kata-kata dari teks sumber diubah menjadi vektor menggunakan *one-hot encoding*, didapat vektor dan dimensi vektor adalah [1, |v|] , seperti Tabel 2 .

Tabel 2. Hasil Encoding Setiap Term dalam Opini

Id-term	Term_Calon	term_Disukai	term_Paling	term_Presiden	term_Rakyat
0	1	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	1	0	0	0
4	0	0	0	0	1

Dari Tabel 2, didapat bahwa *encoder* menghasilkan vektor kata. Misalnya, kata calon menjadi vektor [1 0 0 0 0]. Urutan term dalam matrik hasil *encoding* tidak selalu terurut sesuai urutan kata dalam kalimat *input*. Selanjutnya, vektor w(t) hasil *encoding* tersebut kemudian menjadi vektor *input* dan target *ouput*. W(t) mana yang menjadi *input* dan mana yang menjadi *output* dilihat dari pasangan { *context word, target word*}

Langkah 2: Vektor *input* dari term akan dilewatkan ke *hidden layer* dari sejumlah |v| *neurons*, |v| merupakan jumlah atau ukuran fitur yang digunakan. *Hidden layer* ini merupakan perkalian antara bobot vektor W[|v|, N] dan *input* vector w(t). Matrik bobot W merupakan N adalah jumlah context atau *window*, dan W[|v|, N] akan menjadi *output* (H[1, N]), dimana H adalah *hidden layer*.

Langkah 3: Pada hidden layer tidak terdapat fungsi aktivasi, sehingga H[1,k] akan langsung dilewatkan ke *output* layer.

Langkah 4: *Output* layer dihitung dengan perkalian antara H[1, N] and W'[N, |v|] sehingga menghasilkan vektor U.

Langkah 5: Hitung probabilitas setiap vektor menggunakan fungsi *softmax* dengan Persamaan (1). Setiap iterasi menghasilkan *output* satu nilai *one hot encoding*. Term atau word dengan probabilitas tertinggi adalah target *word* untuk *context word* yang diberikan.

$$p(w_{c,j} = w_{O,c}|w_I) = \frac{\exp u_{c,j}}{\sum_{j'=1}^{V} \exp u_{j'}} \quad (1)$$

Dimana :

$w_{(c,j)}$ adalah kata ke-j yang diprediksi pada posisi *context* ke-c;

$w_{(O,c)}$ adalah kata aktual yang ada pada posisi *context* ke-c;

$w_{(I)}$ adalah kata *input*;

u (c, j) adalah nilai ke-j dalam vektor *U* saat memprediksi kata untuk posisi konteks ke-c.

Ilustrasi hasil akhir Word2Vec dari Tabel 2 adalah matrik vektor dalam Tabel 3. Hasil Tabel 3 berdasarkan parameter *window*=2, ukuran fitur =4, minimum frekuensi kata = 1 (artinya term/ kata yang muncul minimal sekali dalam dokumen akan dilibakan dalam proses pembelajaran), dan *learning rate* (alpha)=0.025. Jika ukuran fitur tidak ditentukan, Skip-gram membuat ukuran fitur 100 secara default[6][15].

Tabel 3. Word *Output* dan Vektor dari Contoh Opini

Kata	V0	V1	V2	V3
Calon	-0.08760262	-0.01231468	0.00916845	-0.0100468
Presiden	-0.09195898	0.04030308	0.04244656	0.05564094
Paling	-0.04840051	0.08610519	-0.02316239	0.0142405
disukai	-0.09956281	0.02452216	-0.01648195	-0.0693512
rakyat	0.06502694	0.05556553	-0.09928826	0.02339865

Pada Tabel 3, kolom v0, v1,v2,v3 menunjukkan fitur yang terbentuk sesuai ukuran fitur. Matrik vektor hasil word2vec merepresentasikan seluruh vektor term dalam korpus yang digunakan.

Langkah 6: Memilih vektor *output* yang akan digunakan sebagai masukan model klasifikasi. Metode yang digunakan adalah *average base*, yaitu mencari nilai rata-rata dari vektor-vektor kata penyusun opini yang akan diprediksi jenis sentimennya.

Contoh:

Opini a: presiden disukai rakyat

Opini b: calon presiden rakyat

Dengan melihat Tabel 2, nilai *average* untuk opini a pada Tabel 4 dan opini b pada Tabel 5. Nilai rata-rata setiap fitur inilah yang akan digunakan sebagai atribut untuk membangun model klasifikasi.

Tabel 4. Word *Output* dan Vektor Opini a

Kata	**V0**	**V1**	**V2**	**V3**
Presiden	-0.09195898	0.04030308	0.04244656	0.05564094
disukai	-0.09956281	0.02452216	-0.01648195	-0.0693512
rakyat	0.06502694	0.05556553	-0.09928826	0.02339865
Rata-rata	**-0,017520524**	**0,038836256**	**-0,017463518**	**0,002776418**

Tabel 5. Word *Output* dan Vektor Opini b

Kata	**V0**	**V1**	**V2**	**V3**
Calon	-0.08760262	-0.01231468	0.00916845	-0.0100468
Presiden	-0.09195898	0.04030308	0.04244656	0.05564094
rakyat	0.06502694	0.05556553	-0.09928826	0.02339865
Rata-rata	**-0,029200873**	**0,02785131**	**-0,015891083**	**0,022997597**

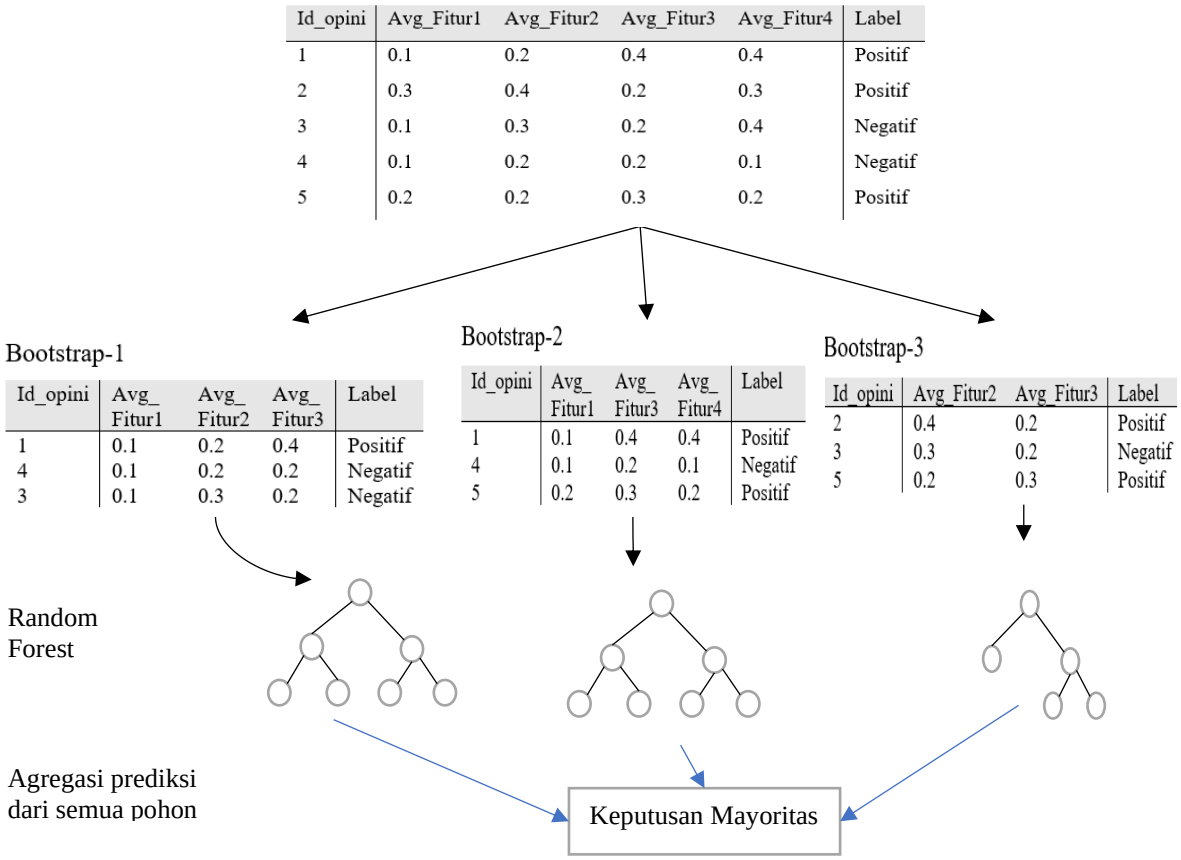
2.4. Model Klasifikasi

Word2Vec akan menghasilkan banyak vektor untuk setiap kata. Machine learning dengan model tunggal seperti *Tree* bisa menghasilkan performa kurang baik karena solusi prediksi hanya tunggal. Konsep *ensemble* mengatasi masalah pencapaian solusi prediksi menjadi optimal karena menghasilkan banyak solusi prediksi. Dengan fungsi konsensus tertentu, prediksi paling optimal dapat diraih. Random Forest adalah salah satu jenis ensemble dari *Tree*.

Jika *Tree* membentuk pohon tunggal dari dataset maka Random Forest (RF) menghasilkan banyak pohon dari dataset. Sebelum membentuk pohon, beberapa sub dataset (*bootstrap*) dibentuk dengan sejumlah N-sampel yang diambil secara random dari dataset asli. Nilai N pada N-sample tersebut harus lebih kecil dari jumlah N dataset asli. Misalnya, dataset asli berisi 5 data dengan empat fitur (Avg_fitur1, Avg_fitur2, Avg_fitur3, Avg_fitur4) dan dua kelas (Positif dan Negatif). RF dibentuk dengan 3 *bootstrap* dengan setiap dataset *bootstrap* ditentukan berisi 3 data. Selanjutnya, setiap *bootstrap* akan membentuk pohon. Berikutnya,

ISSN: 1829-667X / E-ISSN: 2460-9021
DOI: 10.31515/telematika.v18i1.4493

proses pembelajaran akan menentukan solusi prediksi dengan teknik voting atau kelas prediksi mayoritas. Contoh ilustrasi voting, data opini ke-1 diprediksi sebagai kelas positif oleh pohon pertama dan kedua, sedangkan pohon ketiga memprediksi sebagai kelas negatif, maka kelas opini ke-1 menurut Random Forest adalah positif. Ilustrasi cara kerja RF ditunjukkan oleh Gambar 3 (Angka-angka rata-rata fitur, Avg_Fitur1 dan seterusnya hanya sebagai ilustrasi).



Gambar 3. Ilustrasi Random Forest

3. Hasil dan Pembahasan

Tujuan penelitian ini adalah menerapkan Word2Vec sebagai teknik ekstraksi fitur kata dalam model analisis sentimen. Word2Vec mempunyai beberapa parameter, dimana penggunaan nilai parameter yang berbeda-beda berpengaruh pada fitur yang dihasilkan. Eksperimen dalam penelitian ini menggunakan beberapa uji nilai parameter Word2Vec yang berbeda-beda, sehingga didapat fitur-fitur kata terbaik. Skenario setting parameter yaitu pada penggunaan beberapa nilai *epoch* dan *window* yang berbeda-beda. Percobaan dilakukan sebanyak 36 kali. Setiap percobaan *word to vector* dibentuk sesuai parameter Word2Vec dan jumlah data dalam korpus masing-masing. Setiap percobaan menghasilkan 36 dataset vector yang khas. Dataset vector tersebut akan dipakai untuk membentuk 36 model analisis sentimen dengan menggunakan *machine learning* Random Forest.

Tahap pertama, model klasifikasi dibangun menggunakan 26813 komentar yang didapat dari dataset debat 1,2,4,dan 5, sedangkan data uji menggunakan dataset debat 3, sejumlah 5134

komentar (lihat Tabel 1). Beberapa skenario percobaan menggunakan persentase jumlah data *training* dan *testing* pada Tabel 6. Skenario tersebut bertujuan menguji stabilitas model, apakah model konsisten meskipun diuji dengan data sedikit maupun banyak. *Data training* digunakan untuk membangun model klasifikasi dan *data testing* untuk menguji model.

Tabel 6. Pembagian Data untuk Pemodelan dan Pengujian

Percobaan ke-	Data Training	Data Testing
1	10725	2054
2	16088	3080
3	21450	4107
4	26813	5134

Dengan empat percobaan dihasilkan 36 model dengan setiap model mempunyai kombinasi jumlah parameter *window* dan *epoch* berbeda-beda, seperti Tabel 7.

Tabel 7. Akurasi Seluruh Model Berdasarkan Parameter *Window* dan *Epoch*

Model Ke-	Window	Epoch	Data training	Data testing	Akurasi model	Akurasi testing
1	3	1	10725	2054	0,9582	0,9550
2	3	1	16088	3080	0,9281	0,9169
3	3	1	21450	4107	0,8744	0,8532
4	3	1	26813	5134	0,8446	0,8255
5	3	5	10725	2054	0,9679	0,9562
6	3	5	16088	3080	0,9296	0,9177
7	3	5	21450	4107	0,8851	0,8541
8	3	5	26813	5134	0,8474	0,8266
9	3	20	10725	2054	0,9708	0,9562
10	3	20	16088	3080	0,9401	0,9180
11	3	20	21450	4107	0,8832	0,8568
12	3	20	26813	5134	0,8476	0,8290
13	5	1	10725	2054	0,9572	0,9550
14	5	1	16088	3080	0,9238	0,9169
15	5	1	21450	4107	0,8731	0,8532
16	5	1	26813	5134	0,8440	0,8255
17	5	5	10725	2054	0,9689	0,9550
18	5	5	16088	3080	0,9370	0,9173
19	5	5	21450	4107	0,8864	0,8551
20	5	5	26813	5134	0,8515	0,8266
21	5	20	10725	2054	0,9699	0,9556
22	5	20	16088	3080	0,9238	0,9180
23	5	20	21450	4107	0,8834	0,8583
24	5	20	26813	5134	0,8558	0,8294
25	10	1	10725	2054	0,9679	0,9550
26	10	1	16088	3080	0,9242	0,9169
27	10	1	21450	4107	0,8701	0,8532
28	10	1	26813	5134	0,8461	0,8266
29	10	5	10725	2054	0,9747	0,9556
30	10	5	16088	3080	0,9273	0,9188
31	10	5	21450	4107	0,8824	0,8561

Model Ke-	Window	Epoch	Data training	Data testing	Akurasi model	Akurasi testing
32	10	5	26813	5134	0,8487	0,8275
33	10	20	10725	2054	0,9679	0,9562
34	10	20	16088	3080	0,9296	0,9192
35	10	20	21450	4107	0,8914	0,8598
36	10	20	26813	5134	0,8562	0,8298

Rata-rata akurasi berdasarkan jumlah *window* ada pada Tabel 8.

Tabel 8. Rata-Rata Akurasi Model Berdasarkan Parameter *Window*

No	Jumlah *Window*	Rata-rata akurasi model (%)	Rata-rata akurasi testing(%)
1	3	90, 64	88,88
2	5	90, 62	88,88
3	10	90, 72	88,95

Tabel 8 menunjukkan bahwa penggunaan parameter *window* 10 atau kurang, tidak signifikan mempengaruhi peningkatan akurasi model. Tabel 3 justru menunjukkan bahwa model sedikit *overfitting*, karena akurasi pengujian sedikit turun dari akurasi model. Bagaimana jika parameter *epoch* diubah-ubah?. Seperti pada Tabel 9, jumlah *epoch* 20 atau kurang menghasilkan akurasi pada kisaran 90,1 % sampai 91%.

Tabel 9. Rata-Rata Akurasi Model Berdasarkan Parameter *Epoch*

No	*Epoch*	Rata-rata akurasi model (%)	Rata-rata akurasi pengujian model(%)
1	1	90,1	88,77
2	5	90,89	88,89
3	20	91,00	89,05

Parameter *epoch* menunjukkan jumlah iterasi maksimal yang dilakukan oleh proses Word2Vec menuju titik konvergensinya. Tabel 9 menunjukkan peningkatan jumlah akurasi meskipun tidak signifikan, kemungkinan karena jumlah *epoch* kurang tinggi.

4. Kesimpulan dan Saran

Penelitian ini bertujuan mengetahui efek ekstraksi fitur dengan Word2Vec terhadap akurasi model analisis sentimen komentar YouTube yang dibangun dengan Random Forest. Berdasarkan percobaan dengan 1, 5, dan 20 *epoch* serta penggunaan ukuran *window* 3,5, dan 10, didapat rata-rata akurasi model 90,1 % sampai 91%. Namun saat diuji, akurasi model hanya 88,77% sampai dengan 89,05%. Akurasi pengujian model justru turun meskipun tidak signifikan. Selain itu, hasil percobaan menunjukkan jumlah *epoch* dan ukuran *window* mempengaruhi akurasi, semakin tinggi nilai *epoch* dan *window* maka akurasi model menjadi lebih tinggi, meskipun dalam percobaan ini kenaikan akurasi tidak signifikan. Untuk melihat pengaruh *epoch* dan *window* pada Word2Vec, disarankan penggunaan sejumlah *epoch* yang banyak dan ukuran *window* yang lebih besar sehingga konteks kata mempunyai tingkat semantik lebih tinggi.

Daftar Pustaka

[1] A. S. Aribowo, H. Basiron, N. S. Herman, and S. Khomsah, "An evaluation of preprocessing steps and *Tree*-based ensemble machine learning for analysing sentiment on Indonesian youtube comments," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 5, 2020, doi: 10.30534/ijatcse/2020/29952020.

[2] R. D. Handayani, K. Kusrini, and H. Al Fatta, "Perbandingan Fitur Ekstraksi Untuk Klasifikasi Emosi Pada Sosial Media," *Jurnal Ilmiah SINUS*, vol. 18, no. 2, p. 21, 2020, doi: 10.30646/sinus.v18i2.457.

[3] G. A. Dalaorao, A. M. Sison, and R. P. Medina, "Integrating Collocation as TF-IDF Enhancement to Improve Classification Accuracy," *TSSA 2019 - 13th International Conference on Telecommunication Systems, Services, and Applications, Proceedings*, pp. 282–285, 2019, doi: 10.1109/TSSA48701.2019.8985458.

[4] S. Qaiser and R. Ali, "Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents," *International Journal of Computer Applications*, vol. 181, no. 1, pp. 25–29, 2018, doi: 10.5120/ijca2018917395.

[5] K. Ethayarajh, D. Duvenaud, and G. Hirst, "Understanding undesirable word embedding associations," *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pp. 1696–1705, 2020, doi: 10.18653/v1/p19-1166.

[6] X. Rong, "word2vec Parameter Learning Explained," pp. 1–21, 2014, [Online]. Available: http://arxiv.org/abs/1411.2738.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, pp. 1–12, 2013.

[8] M. A. Fauzi, "Word2Vec model for sentiment analysis of product reviews in Indonesian language," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 1, p. 525, 2019, doi: 10.11591/ijece.v9i1.pp525-530.

[9] E. M. Alshari, A. Azman, S. Doraisamy, N. Mustapha, and M. Alkeshr, "Effective Method for Sentiment Lexical Dictionary Enrichment Based on Word2Vec for Sentiment Analysis," *Proceedings - 2018 4th International Conference on Information Retrieval and Knowledge Management: Diving into Data Sciences, CAMP 2018*, pp. 177–181, 2018, doi: 10.1109/INFRKM.2018.8464775.

[10] X. Yang, C. Macdonald, and I. Ounis, "Using word embeddings in Twitter election classification," *Information Retrieval Journal*, vol. 21, no. 2–3, pp. 183–207, 2018, doi: 10.1007/s10791-017-9319-5.

[11] N. Cahyana, S. Khomsah, and A. S. Aribowo, "Improving Imbalanced Dataset Classification Using Oversampling and Gradient Boosting," in *Proceeding - 2019 5th International Conference on Science in Information Technology: Embracing Industry 4.0: Towards Innovation in Cyber Physical System, ICSITech 2019*, 2019, pp. 217–222, doi: 10.1109/ICSITech46713.2019.8987499.

[12] A. S. Aribowo, Y. Fauziah, H. Basiron, and N. S. Herman, "Proceedings of the 2 nd Faculty of Industrial Technology International Congress International Conference Clustering Emotional Features using Machine Learning in Public Opinion during the 2019 Presidential Candidate Debates in Indonesia," vol. 6, pp. 2–7, 2020.

[13] S. Khomsah and A. S. Aribowo, "Model Text-Preprocessing Komentar Youtube Dalam Bahasa Indonesia," vol. 1, no. 10, pp. 1–8, 2021.

[14] C. Mccormick, "Word2Vec Tutorial - The Skip-Gram Model," 2016.

[15] R. P. Nawangsari, R. Kusumaningrum, and A. Wibowo, "Word2vec for Indonesian sentiment analysis towards hotel reviews: An evaluation study," *Procedia Computer Science*, vol. 157, pp. 360–366, 2019, doi: 10.1016/j.procs.2019.08.178.

[16] "Skip-Gram: NLP context words prediction algorithm" ,https://towardsdatascience.com/skip-gram-nlp-context-words-prediction-algorithm-5bbf34f84e0c.