Self-Training Naive Bayes Berbasis *Word2Vec* untuk Kategorisasi Berita Bahasa Indonesia

Joan Santoso[1,2], Agung Dewa Bagus Soetiono[2], Gunawan[2], Endang Setyati[2], Eko Mulyanto Yuniarno[3,4], Mochamad Hariadi[3,4], Mauridhi Hery Purnomo[3,4]

***Abstract*—News as one kind of information that is needed in daily life has been available on the internet. News website often categorizes their articles to each topic to help users access the news more easily. Document classification has widely used to do this automatically. The current availability of labeled training data is insufficient for the machine to create a good model. The problem in data annotation is that it requires a considerable cost and time to get sufficient quantity of labeled training data. A semi-supervised algorithm is proposed to solve this problem by using labeled and unlabeled data to create classification model. This paper proposes semi-supervised learning news classification system using Self-Training Naive Bayes algorithm. The feature that is used in text classification is Word2Vec Skip-Gram Model. This model is widely used in computational linguistics or text mining research as one of the methods in word representation. Word2Vec is used as a feature because it can bring the semantic meaning of the word in this classification task. The data used in this paper consists of 29,587 news documents from Indonesian online news websites. The Self-Training Naive Bayes algorithm achieved the highest F1-Score of 94.17%.**

***Intisari*—Berita sebagai salah satu jenis informasi yang dibutuhkan dalam kehidupan sehari-hari telah tersedia secara bebas di internet. Situs berita telah melakukan pengelompokan berita berdasarkan topiknya untuk mempermudah pengguna mencari berita yang dibutuhkan. Klasifikasi dokumen telah banyak digunakan untuk membantu pengelompokan berita secara otomatis. Kurang tersedianya data pelatihan yang cukup untuk digunakan komputer membentuk model klasifikasi yang baik sering menjadi kendala dalam implementasi di kasus nyata. Masalah utama dalam pelabelan data pelatihan agar diperoleh jumlah data yang cukup adalah perlunya biaya yang besar dan waktu yang cukup lama. Algoritme *semi-supervised* telah ditawarkan untuk menjawab permasalahan tersebut dengan menggunakan data berlabel dan tak berlabel dalam membentuk model klasifikasi yang dibutuhkan. Makalah ini mengusulkan sistem klasifikasi berita menggunakan *semi-supervised learning* dengan algoritme *Self-Training Naive Bayes*. Fitur yang digunakan dalam klasfikasi teks ini adalah model *Word2Vec Skip-Gram*. Model ini banyak digunakan untuk merepresentasikan kata dalam penelitian terbaru di bidang linguistik komputasi atau *text mining*. Alasan utama digunakannya *Word2Vec* adalah dapat menambahkan makna semantik dari kata dalam proses klasifikasi. Data yang digunakan dalam klasifikasi ini adalah data berita bahasa Indonesia dengan jumlah 29.587 dokumen. Hasil percobaan *Self-Training Naive Bayes* memiliki nilai *F1-Score* terbaik sebesar 94,17%.**

***Kata Kunci*— Kategorisasi Berita, *Word2Vec*, *Skip-Gram*, *Self-Training*, *Naive Bayes*, *Semi-supervised Learning*, Bahasa Indonesia.**

I. PENDAHULUAN

Perkembangan teknologi internet di Indonesia telah menyebabkan pertumbuhan jumlah data dan pengguna. Di situs Statistia pada tahun 2015 pengguna internet tercatat 34,61%, meningkat menjadi 39,7% di tahun 2017, dari total jumlah penduduk di Indonesia [1].

Salah satu informasi yang paling banyak dicari adalah berita. Teknologi internet dapat menyebabkan berita dapat disampaikan lebih mudah. Munculnya banyak situs berita yang ada di Indonesia menyebabkan pertumbuhan berita yang cepat, banyak, dan tidak terkelompok. Situs *news aggregator* saat ini telah banyak berkembang dan telah secara otomatis melakukan pengelompokan berita berdasarkan topik yang sesuai [2]. Kategorisasi berita tersebut dibantu oleh komputer dengan algoritme klasifikasi dokumen.

Klasifikasi dokumen dalam *supervised learning* membutuhkan data pelatihan yang cukup untuk mendapatkan hasil yang baik. Jumlah data pelatihan yang tersedia tidak mencukupi jika dibandingkan dengan cepatnya pertumbuhan data saat ini [3]. Hal ini disebabkan oleh proses pelabelan data pelatihan yang membutuhkan biaya besar dan waktu relatif lama [4]. Metode *semi-supervised learning* dikembangkan untuk menjawab permasalahan tersebut.

Semi-supervised learning menggunakan data pelatihan yang berlabel dan data tak berlabel dalam proses pembelajaran dalam algoritme klasifikasi. Data tak berlabel dapat diperoleh dengan mudah dan dipercaya dapat membantu algoritme klasifikasi untuk menghasilkan model klasifikasi yang baik. Pendekatan *semi-supervised learning* telah banyak digunakan di bidang *text mining* dan klasifikasi teks [3], [5].

[1] *Mahasiswa, Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember, Jln. Teknik Mesin, Gedung B, C, dan AJ, Kampus ITS, Sukolilo, Surabaya 60111, Jawa Timur, Indonesia (tlp: 031-5994251, fax : 031-5931237, e-mail: joan.santoso13@mhs.ee.its.ac.id).*

[2] *Dosen, Departemen Teknologi Informasi, Sekolah Tinggi Teknik Surabaya, Jln. Ngagel Jaya Tengah 73-77, Surabaya 60284, Jawa Timur, Indonesia (tlp: 031-5027920, fax: 031-5041509, e-mail: joan@stts.edu,agungdewa@stts.edu, gunawan@ stts.edu, esetyati @stts.edu).*

[3] *Dosen, Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember, Jln. Teknik Mesin, Gedung B, C, dan AJ, Kampus ITS, Sukolilo, Surabaya 60111, Jawa Timur, Indonesia (tlp: 031-5994251, fax : 031-5931237, e-mail: ekomulyanto@ee.its.ac.id, mochar@ee.its.ac.id, hery@ee. its.ac.id).*

[4] *Dosen, Departemen Teknik Komputer, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember, Jln. Teknik Mesin, Gedung B dan C, Kampus ITS, Sukolilo, Surabaya 60111, Jawa Timur, Indonesia (tlp: 031-5922936, e-mail: ekomulyanto@ ee.its.ac.id, mochar@ee.its.ac.id, hery@ee.its.ac.id).*

Fokus makalah ini adalah klasifikasi topik berita bahasa Indonesia dengan metode *semi-supervised learning*, yaitu *Self-Training*. Algoritme klasifikasi yang digunakan dalam makalah ini adalah *Naive Bayes*. *Naive Bayes* umum digunakan dalam klasifikasi teks dan memiliki kinerja yang cukup baik [6]. Fitur yang digunakan dalam klasifikasi dokumen berita adalah *Word2Vec Skip-Gram*. Model ini digunakan sebagai fitur dalam klasifikasi karena dipercaya dapat menambahkan makna semantik dari kata dalam proses klasifikasi [7].

Penjabaran dari detail makalah ini akan dibagi menjadi lima bagian. Bagian pertama menjabarkan pendahuluan dan latar belakang. Bagian kedua menjabarkan penelitian terkait dengan kategorisasi teks. Bagian ketiga dan keempat menjabarkan metode klasifikasi teks dan hasil uji coba, sedangkan bagian kelima berisi kesimpulan.

II. APLIKASI-APLIKASI KATEGORISASI TEKS

Kategorisasi teks telah diaplikasikan secara luas pada beberapa bidang di dunia, antara lain kategorisasi artikel sebuah situs dan opini dari seseorang. Sebuah penelitian melakukan klasifikasi teks artikel situs bahasa China [8]. Penelitian lain mengkategorikan tulisan opini seseorang dalam bahasa Inggris menjadi positif atau negatif [9].

Penelitian yang lalu telah menggunakan berbagai pendekatan metode, di antaranya *supervised learning*, *unsupervised learning*, dan *semi-supervised learning*. Klasifikasi emosi dari teks dengan metode *supervised learning*, yaitu *k-Nearest Neighbour* (k-NN) telah dilakukan [10]. Penelitian lain melakukan kategorisasi teks dengan menggunakan struktur laten dan metode *unsupervised learning* dengan algoritme *Self Organizing Map* [11]. Klasifikasi teks dengan pendekatan *semi-supervised learning Co-Training* untuk dokumen *website* akademik dan *email* juga telah dilakukan [12], [13]. Penelitian lainnya telah membentuk model klasifikasi teks menggunakan teks dari sebuah situs sebagai data pelatihan dengan pendekatan *Self-Training* [14].

Penelitian kategorisasi teks di bahasa Indonesia diterapkan di berbagai bidang, di antaranya adalah klasifikasi keluhan di sebuah instansi pemerintah [15], klasifikasi berita bahasa Indonesia [16], dan identifikasi *personality* dari *tweet* seseorang di sosial media Twitter [17]. Penggunaan beberapa jenis algoritme *machine learning* untuk klasifikasi dokumen di bahasa Indonesia telah diusulkan dalam penelitian sebelumnya. *Naive Bayes* digunakan untuk melakukan klasifikasi artikel Wikipedia Bahasa Indonesia untuk menentukan artikel terkait dengan hewan atau tumbuhan dalam pembentukan *ontology* [6]. Algoritme *Support Vector Machine* (*SVM*) telah diguanakan untuk melakukan klasifikasi pesan di sosial media menjadi empat kategori, yaitu netral, positif, negatif, dan pertanyaan [18].

Fitur yang umum digunakan dalam klasifikasi teks adalah *bag-of-words*, tetapi saat ini *word vector* diusulkan sebagai cara baru untuk merepresentasikan kata di bidang pengolahan bahasa alami. *Word vector* dibentuk dengan bantuan *Neural Network*. Teknik ini diusulkan dengan nama *Word2Vec* [19]. Terdapat dua buah model yang diusulkan, yaitu *Continuous Bag-of Words (CBOW)* dan *Skip-Gram*.

Pemanfaatan *Word2Vec* sebagai fitur dalam klasifikasi dokumen di bahasa Inggris telah diusulkan oleh dua peneliti [7], [20]. Hasil penggunaan *Word2Vec* sebagai fitur di penelitian yang lalu memiliki kinerja lebih baik jika dibandingkan dengan teknik representasi yang lain [20]. Penggunaan *Word2Vec* untuk klasifikasi dokumen berita di bahasa Indonesia juga telah diusulkan [2]. Hasil percobaan klasifikasi di bahasa Indonesia memberikan angka *F1-Score* terbaik, yaitu 81,10% dengan model yang digunakan adalah *Skip-Gram* dengan dimensi 200.

Penelitian terkait dengan kategorisasi teks di bahasa Indonesia telah banyak dilakukan dengan metode *supervised* dan *unsupervised learning*. Kebutuhan jumlah data pelatihan yang cukup menjadi masalah utama sehingga diusulkan metode *semi-supervised learning* untuk klasifikasi berita bahasa Indonesia pada makalah ini. Usulan metode *semi-supervised learning* yang dipilih adalah algoritme *Self-Training* [14] dan representasi fitur yang digunakan adalah *Word2Vec* dengan *average pooling* [20].

Kontribusi yang diberikan pada makalah ini adalah diusulkannya metode *semi-supervised learning* dalam klasifikasi berita bahasa Indonesia. Metode yang ditawarkan memiliki kelebihan, yaitu sedikitnya penggunaan jumlah data pelatihan berlabel jika dibandingkan dengan pendekatan-pendekatan *supervised learning* pada penelitian sebelumnya.

III. KLASIFIKASI BERITA DENGAN ALGORITME *SELF-TRAINING NAIVE BAYES*

Proses dari penelitian ini dibagi menjadi tiga subproses, yaitu *preprocessing* atau pembentukan daftar kata, pembentukan vektor dokumen sebagai fitur klasifikasi, dan pembentukan model atau klasifikasi topik berita dengan algoritme *Self-Training Naive Bayes*. Urutan langkah setiap proses secara runut diperlihatkan pada Gbr. 1.

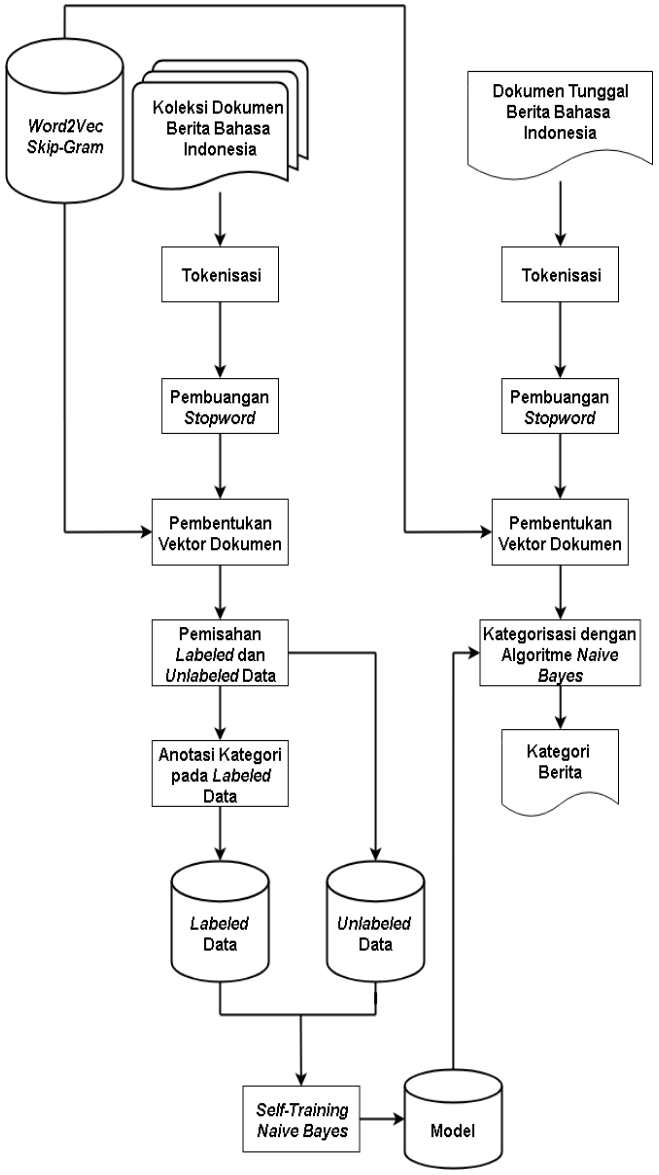
A. Preprocessing untuk Mendapatkan Daftar Kata di Dokumen

Hingga penelitian ini selesai dilakukan belum ditemukan data publik yang telah divalidasi dan dapat digunakan sebagai standardisasi data untuk penelitian kategorisasi berita di bahasa Indonesia. Oleh sebab itu, makalah ini menggunakan data dari sumber yang sama dengan penelitian sebelumnya [21]. Jumlah *data set* yang digunakan adalah 29.587 berita dari situs berita CNN Indonesia yang terdiri atas lima kategori topik, yaitu teknologi, hiburan, ekonomi, olahraga, dan politik.

Proses akuisisi berita dibantu dengan sebuah *crawler* sederhana. Ekstraksi isi berita dari setiap halaman situs berita dibantu dengan *rule* dan *regular expression*. Untuk mendapatkan daftar kata dari sebuah halaman situs, dilakukan beberapa proses sebagai berikut.

1) Tokenisasi: Proses tokenisasi pada dokumen dilakukan dengan bantuan *regular expression*. Tujuan proses ini adalah untuk mengeliminasi karakter-karakter tanda baca atau nonalfanumerik sehingga didapatkan hanya sekumpulan kata-kata yang terdapat di dalam dokumen. Proses ini dilengkapi dengan *case folding* untuk mengubah seluruh kata atau token dalam bentuk yang sama, yaitu huruf kecil.

2) Pembuangan Stopword: Proses ini dilakukan untuk mengeliminasi kata-kata yang sering muncul di dalam dokumen. Daftar *stopword* yang digunakan diperoleh dari Talla [22].



Gbr. 1 Alur proses klasifikasi teks dengan *self-training.*

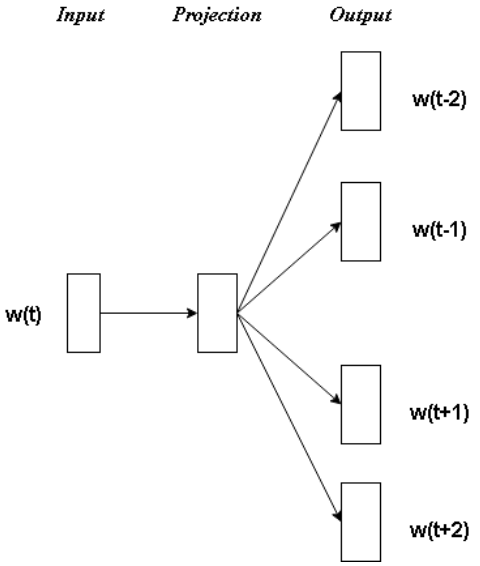
B. Pembentukan Vektor Dokumen sebagai Fitur Klasifikasi

Kelemahan penggunaan *bag-of-words* sebagai fitur klasifikasi teks adalah ukuran/dimensi yang besar. Jumlah fitur yang besar dapat menurunkan kinerja dari algoritme klasifikasi yang digunakan, karena tidak semua fitur tersebut relevan [23].

Word vector diusulkan sebagai salah satu metode untuk merepresentasikan kata dengan kelebihan strukturnya yang padat dan dipercaya dapat memberikan makna semantik. *Word2Vec* merupakan model yang populer digunakan, terdiri atas dua jenis, yaitu *Continuous Bag-of Words* (CBOW) dan *Skip-Gram.* Beda antara *Skip-Gram* dan CBOW terletak pada arsitektur *Neural Network* dan cara pembentukan model dari masing-masing jenis. *Skip-Gram* memberikan masukan kata dan meminta *Neural Network* untuk menebak kata-kata yang muncul di sekitar kata masukan. Untuk arsitektur CBOW, *Neural Network* digunakan untuk menebak sebuah kata dengan masukan adalah kata sekitar dari kata yang ditebak.

Model *word2vec* yang digunakan adalah *Skip-Gram*. *Skip-Gram* memiliki kelebihan yaitu lebih sesuai untuk kata-kata yang jarang muncul dibandingkan CBOW [24]. Alasan mendasar pemilihan model *Skip-Gram* adalah terdapat variasi kata dengan jumlah yang banyak, sehingga dikhawatirkan akan terjadi kemunculan kata yang jarang dijumpai. Penggunaan *Skip-Gram* pada permasalahan ini dinilai lebih tepat dibandingkan dengan CBOW.

Arsitektur dari *Skip-Gram* ditunjukkan pada Gbr. 2, yaitu terdiri atas sebuah lapisan tersembunyi *Neural Network*. Pembentukan model *Skip-Gram* dipengaruhi oleh beberapa parameter, di antaranya jumlah dimensi dari neuron (yang disebut dengan variabel *d*) dan jumlah *window* kata (yang disebut dengan variabel *t*) yang digunakan.



Gbr. 2 Ilustrasi model *Neural Network* untuk *Word2Vec Skip-Gram* [19].

Jika sebuah dokumen dilambangkan dengan *D* dan setiap kata yang ada di dalam dokumen tersebut dilambangkan dengan w_i dengan *i* merupakan indeks kata di dalam dokumen, dan *N* adalah jumlah kata yang ada di dalam dokumen, maka sebuah dokumen dapat dimodelkan dengan $D=\{w_1,w_2,w_3,...,w_N\}$. Proses pelatihan *Neural Network* dari model *Skip-Gram* jika diberikan nilai *window* t = 3, maka untuk setiap w_i akan digunakan sebagai masukan dari *Neural Network* untuk menebak sekumpulan kata dari $w_{t-i},...,w_{t+i,}$ yaitu $w_{i-3},...,w_{i+3.}$ Fungsi objektif yang digunakan dalam pelatihan *Neural Network* dituliskan pada (1) dan (2).

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} \sum_{-t \le j \le t, j \ne 0} log\, p(w_{t+j}|w_t) \quad (1)$$

$$p(w_j|w_i) = \frac{e^{(v'_{wj}{}^T v_{wi})}}{\sum_{k=1}^{m} e^{(v'_{wk}{}^T v_{wi})}}. \quad (2)$$

Variabel v_w dan v'_w adalah representasi vektor masukan dan keluaran dari sebuah kata *w*. Variabel *m* adalah jumlah kata

unik yang ada di daftar kosakata. *One-hot encoding* digunakan sebagai representasi kata dari masukan dan keluaran menjadi sebuah vektor. Vektor *one-hot encoding* tersebut memiliki ukuran/dimensi sejumlah kata unik yang ada di dalam kumpulan dokumen dan akan bernilai satu pada indeks kata tersebut.

Dua model optimasi pada pembentukan *Word2Vec* adalah *hierarchical softmax* dan *negative sampling*. *Negative sampling* merupakan algoritme optimasi yang lebih efisien dibandingkan *hierarchical softmax* [24]. Pertimbangan tersebut menyebabkan optimasi *negative sampling* digunakan dalam makalah ini.

Pembentukan model *Skip-Gram* pada makalah ini menggunakan parameter jumlah *window* adalah 10. Jumlah data yang digunakan adalah 308.227 artikel dengan 295.485 kata unik dari artikel Wikipedia bahasa Indonesia yang telah diproses menjadi teks bersih. Ukuran/dimensi *Word2Vec* yang dihasilkan beraneka ragam, mulai dari 100 sampai dengan 500.

Model hasil pelatihan *Neural Network* adalah sebuah matriks M_{emb} dengan ukuran sebesar *m* x *d*. Variabel *d* adalah jumlah dimensi vektor kata dari *Word2Vec*. Untuk mendapatkan sebuah vektor dari kata, vektor *one-hot* v_w dari sebuah kata *w* dikalikan dengan matriks M_{emb}. Persamaan (3) menjabarkan cara vektor kata, yang dilambangkan dengan variable $v_{embedding}$, diperoleh.

$$v_{embedding} = {v_w}^T * M_{emb}. \quad (3)$$

Fitur yang digunakan dalam klasifikasi dokumen dalam makalah ini adalah nilai rata-rata setiap elemen vektor kata yang diperoleh dari *Word2Vec* [20]. Persamaan untuk mendapatkan vektor dokumen dituliskan melalui (4).

$$emb_i = \frac{\sum_{j=1}^{n} v_j^i}{n}. \quad (4)$$

Variabel emb_i adalah nilai vektor dokumen untuk dimensi ke-*i*. Variabel *n* bernilai jumlah kata yang ada di dalam dokumen. v_i^i adalah isi elemen ke-*i* dari representasi vektor kata ke-*j*. Dari (4) tersebut didapatkan sebuah vektor dokumen $v_{doc} = [emb_1, emb_2, emb_3, \ldots, emb_d]$ sebagai model representasi dari sebuah dokumen berita.

Kelemahan penggunaan *Word2Vec* terletak pada model vektor kata yang dihasilkan hanya terbatas untuk daftar kata yang muncul di dokumen pelatihan. Untuk mengatasi kata yang tidak terdapat dalam daftar kosakata di *Word2Vec*, maka setiap kata tersebut digantikan dengan vektor sepanjang dimensi modelnya yang seluruh elemennya bernilai nol.

C. Algoritme Self-Training Naive Bayes

Algoritme yang digunakan dalam klasifikasi dokumen di makalah ini disebut *Self-Training Naive Bayes*. Nama tersebut diambil dari penggabungan dua buah algoritme, yaitu *Naive Bayes* dan *Self-Training*. *Self-Training* digunakan untuk membantu *Naive Bayes* membentuk model klasifikasi dengan memanfaatkan data pelatihan berlabel dan data tak berlabel.

Self-Training merupakan algoritme *semi-supervised learning* yang umum dan sederhana di *semi-supervised learning* [25]. Masukan dari algoritme *Self-Training* adalah sebuah data yang dibagi menjadi dua, yaitu $L=\{(x_i,y_i)\}_{i=1}^{N}$ dan $U=\{(x_j)\}_{j=1}^{M}$. *L* merupakan data yang diberi label, sedangkan *U* adalah data tidak berlabel. Nilai *N* merupakan jumlah data yang telah diberi label dan nilai *M* merupakan jumlah data yang tidak berlabel. *M* pada umumnya bernilai lebih besar dari *N*. Kerangka umum dari algoritme *Self-Training* yang digunakan diperlihatkan pada Gbr. 3.

Function Self-Training(L:Labeled Dataset, U:Unlabelled Dataset)
1. C = train(L)
2. **WHILE** StopCriteria <> TRUE **DO**
3. L = L + select(label(U, C))
4. C = train(L)
5. **END WHILE**
6. **RETURN** C

Gbr. 3 Kerangka algoritme *Self-Training.*

Cara kerja algoritme *Self-Training* dimulai dengan belajar dari data *L* dan berupaya untuk melabeli data *U*. Sebagian data *U* dipilih dan digabungkan dengan data *L* yang telah dilabeli secara manual sebagai data pelatihan dari *classifier C* pada setiap iterasi. Data *U* yang dilabeli oleh *classifier C* sering kali mengalami kesalahan pelabelan sehingga perlu adanya pemilihan data *U* yang memiliki nilai *confidence* hasil klasifikasi di atas batas ambang yang disebut dengan ε.

Nilai dari batas ambang *confidence* tersebut ditentukan sebelum algoritme *Self-Training* bekerja. Proses ini dilakukan dengan bantuan fungsi *select* di baris ketiga pada Gbr. 3. Untuk menghitung *confidence value* dari hasil pelabelan data *U*, digunakan rumus yang dituliskan pada (5).

$$confidence(x, c_i) = \frac{e^{(P(x|c_i)\,P(c_i))}}{\sum_{j=i}^{C} e^{(P(x|c_j)\,P(c_j))}}. \quad (5)$$

Nilai *confidence* dari sebuah data *x* terhadap sebuah kelas c_i didapatkan dari hasil normalisasi *softmax* dari nilai hasil perhitungan probabilitas posterior, yaitu $P(c|x)=p(x|c).p(c)$ agar diperoleh nilai rentang 0 sampai 1. Jika nilai sebuah *confidence* sebuah data *x* di kelas c_i di atas nilai ε, maka data tersebut akan dipilih.

Proses pembentukan model pada algoritme *Self-Training* dilakukan secara iteratif hingga kondisi berhenti dipenuhi. Kondisi berhenti yang digunakan pada makalah ini adalah iterasi tidak melebihi batas iterasi yang ditentukan atau proses pelabelan data *U* sudah konvergen. Batas iterasi yang ditentukan adalah 200. Algoritme dikatakan konvergen jika hasil prediksi dari data *U* dari iterasi sebelumnya sama dengan hasil predisksi data *U* dari iterasi saat ini.

Algoritme *Naive Bayes* sebagai *classifier* bekerja dengan mencari nilai maksimum dari *probabilitas posterior* kelas target jika diberikan sebuah data *X*. Model matematis dari algoritme *Naive Bayes* ini seperti pada (6).

$$\hat{y} = \underset{c}{argmax}\, P(X|c)P(c) \quad (6)$$

dengan *X* melambangkan data yang diklasifikasi, sedangkan *c* adalah kelas target yang dicari. *P*(*c*) mendefinisikan nilai

probabilitas *prior* dari kelas *c* dan *P*(*X*|*c*) mendefinisikan probabilitas data *X* jika diketahui targetnya adalah *c*.

Fitur yang digunakan adalah data kontinu, sehingga perhitungan *conditional probability* di algoritme *Naive Bayes* ini menggunakan fungsi *kernel* Gaussian. Persamaan untuk mendapatkan nilai *conditional probability* dituliskan dalam (7) dan (8).

$$P(X|C) = \prod_{i=1} P(X_i|C) \quad (7)$$

$$P(X_i|C) = \frac{1}{\sqrt{2\pi\,\sigma_C^2}} e^{\frac{-\left(X_i-\mu_c\right)^2}{2\sigma_C^2}}. \quad (8)$$

Variabel μ_c mendefinisikan rata-rata nilai data *Xi* pada kelas *C*. Variabel σ_c^2 adalah varian nilai dari *Xi* di kelas *C*. Sedangkan untuk perhitungan probabilitas *prior* kelas *C* diperoleh dari (9).

$$P(C) = \frac{Nc}{N}. \quad (9)$$

Variabel *Nc* merupakan jumlah kelas *C* pada data latih, sedangkan *N* adalah jumlah data latih.

IV. HASIL UJI COBA

Uji coba yang dilakukan pada penelitian dalam makalah ini dibagi menjadi tiga percobaan. Percobaan pertama dilakukan untuk melihat dimensi terbaik dari *Word2Vec* yang digunakan. Percobaan kedua dilakukan untuk mengukur kinerja dari algoritme *Self-Training Naive Bayes* dalam melakukan klasifikasi teks berita bahasa Indonesia. Pada percobaan ketiga dilakukan evaluasi kinerja algoritme *Self-Training Naive Bayes* jika dibandingkan dengan penelitian yang telah dilakukan sebelumnya.

Dalam kategorisasi teks, *F1-Score* merupakan pengukuran kinerja yang sering digunakan [26]. *F1-Score* merupakan rata-rata harmonik dari *Precision* dan *Recall*. *F1-Score* memberikan gambaran seberapa presisi dan seberapa baik model dalam memberikan label klasifikasi. Oleh sebab itu, *F1-Score* digunakan sebagai metode pengukuran kinerja algoritme yang ditawarkan pada makalah ini.

A. Hasil Pengamatan Ukuran Dimensi Word2Vec Skip-Gram terhadap Proses Klasifikasi

Dimensi terbaik akan dilihat dengan menggunakan algoritme *Naive Bayes* dengan metode pengujian *cross validation*. Ukuran/dimensi *Word2Vec* yang digunakan adalah 100, 200, 300, 400, dan 500. Nilai *k* yang digunakan adalah 5, 10, 15, 20, dan 25. Hasil percobaan disajikan pada Tabel I sampai Tabel V.

Hasil terbaik percobaan dengan *Word2Vec* dimensi 100 di Tabel I diperoleh melalui *K*=25 dengan nilai *Recall*, *Precision*, dan *F1-Score* adalah 94,163%, 94,281%, dan 94,173%. Sedangkan rata-rata nilai *Recall*, *Precision*, dan *F1-Score* dari hasil percobaan yang dilakukan adalah 94,157%, 94,265%, dan 94,168%.

Percobaan pada *Word2Vec* dengan ukuran 200 di Tabel II memiliki kinerja terbaik ketika pengujian *cross validation* dilakukan dengan nilai *K*=15, dengan nilai *Recall* 94,315%, nilai *Precision* 94,415%, dan nilai *F1-Score* 94,322%. Nilai rata-rata dari seluruh percobaan tersebut berada pada angka 94,305% untuk *Recall* dan 94,406% untuk *Precision*, sedangkan untuk *F1-Score* adalah 94,312%.

TABEL I
HASIL PERCOBAAN *K-FOLD CROSS VALIDATION* UNTUK UKURAN/DIMENSI *WORD2VEC* 100

No.	Nilai K	*Recall*	*Precision*	*F1-Score*
1	5	94,153%	94,253%	94,164%
2	10	94,160%	94,262%	94,170%
3	15	94,153%	94,259%	94,164%
4	20	94,156%	94,268%	94,167%
5	25	94,163%	94,281%	94,173%
Rata-rata		94,157%	94,265%	94,168%

TABEL II
HASIL PERCOBAAN *K-FOLD CROSS VALIDATION* UNTUK UKURAN/DIMENSI *WORD2VEC* 200

No.	Nilai K	*Recall*	*Precision*	*F1-Score*
1	5	94,288%	94,385%	94,296%
2	10	94,308%	94,407%	94,316%
3	15	94,315%	94,415%	94,322%
4	20	94,308%	94,415%	94,316%
5	25	94,305%	94,410%	94,312%
Rata-rata		94,305%	94,406%	94,312%

Percobaan dengan dimensi sebesar 300 di Tabel III menghasilkan kinerja terbaik dengan nilai *K*=5 dengan nilai *Recall* 94,234%, nilai *Precision* 94,328%, dan nilai *F1-Score* 94,241%. Untuk hasil percobaan secara keseluruhan diperoleh nilai rata-rata kinerja 94,224% untuk *Recall*, 94,326% untuk *Precision*, dan 94,232% untuk *F1-Score*.

TABEL III
HASIL PERCOBAAN *K-FOLD CROSS VALIDATION* UNTUK UKURAN/DIMENSI *WORD2VEC* 300

No.	Nilai K	*Recall*	*Precision*	*F1-Score*
1	5	94,234%	94,328%	94,241%
2	10	94,220%	94,319%	94,229%
3	15	94,220%	94,323%	94,229%
4	20	94,224%	94,330%	94,232%
5	25	94,220%	94,330%	94,229%
Rata-rata		94,224%	94,326%	94,232%

TABEL IV
HASIL PERCOBAAN *K-FOLD CROSS VALIDATION* UNTUK UKURAN/DIMENSI *WORD2VEC* 400

No.	Nilai K	*Recall*	*Precision*	*F1-Score*
1	5	94,204%	94,318%	94,210%
2	10	94,224%	94,340%	94,231%
3	15	94,220%	94,340%	94,228%
4	20	94,214%	94,337%	94,221%
5	25	94,220%	94,349%	94,227%
Rata-rata		94,216%	94,337%	94,223%

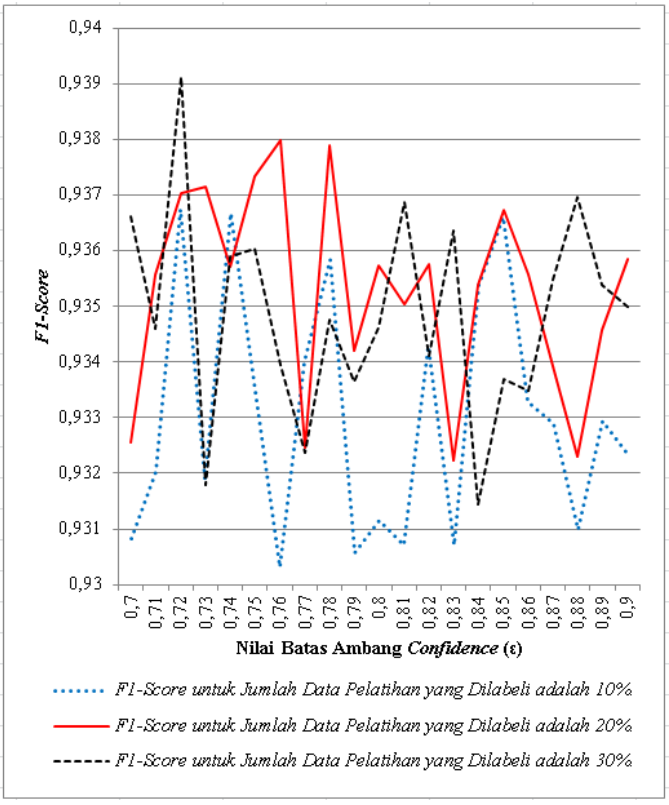
Pada percobaan *Word2Vec* dengan dimensi 400 di Tabel IV, kinerja terbaik diperoleh pada pengujian *cross validation* di nilai *K*=10 dengan nilai *Recall* 94,224%, nilai *Precision* 94,34%, dan nilai *F1-Score* 94,231%. Rata-rata hasil percobaan diperoleh pada nilai *Recall* adalah 94,216%, nilai *Precision* adalah 94,337%, dan nilai *F1-Score* adalah 94,223%.

TABEL V
HASIL PERCOBAAN *K-FOLD CROSS VALIDATION* UNTUK UKURAN DIMENSI *WORD2VEC* 500

No.	Nilai K	*Recall*	*Precision*	*F1-Score*
1	5	94,302%	94,408%	94,311%
2	10	94,322%	94,433%	94,332%
3	15	94,308%	94,429%	94,318%
4	20	94,315%	94,435%	94,324%
5	25	93,12%	94,440%	94,320%
Rata-rata		94,300%	94,415%	94,309%

Percobaan terakhir dilakukan menggunakan *Word2Vec* dengan dimensi 500. Kinerja terbaik di Tabel V diperoleh pada saat pengujian *cross validation* dengan nilai *K*=10. Hasil yang diperoleh dari pengujian tersebut adalah nilai *Recall* pada angka 94,322%, nilai *Precision* pada angka 94,433%, dan nilai *F1-Score* pada angka 94,332%. Nilai rata-rata kinerja dari hasil percobaan ini adalah 94,3% untuk *Recall*, 94,415% untuk *Precision*, dan 94,309% untuk *F1-Score*.

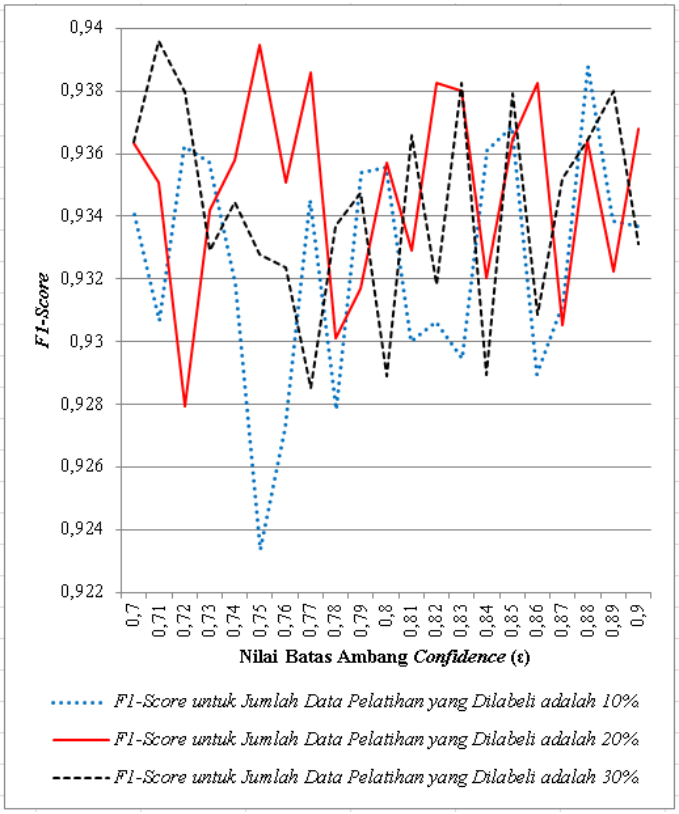
Nilai rata-rata dari pengujian *cross validation* terbaik diperoleh dengan dimensi *Word2Vec* sebesar 200. Nilai rata-rata *Recall* berada pada angka 94,305%, sedangkan nilai rata-rata *Precision* pada angka 94,406%, dan nilai rata-rata *F1-Score* sebesar 94,312%.



Gbr. 4 Grafik nilai *F1-Score* untuk algoritme *Self-Training Naive Bayes* dengan *percentage-splitting* 70%-30%.

B. Hasil Percobaan Self-Training Naive Bayes di Klasifikasi Berita Bahasa Indonesia

Percobaan *Self-Training Naive Bayes* menggunakan pengujian *percentage-splitting*. Data yang ada dibagi menjadi tiga proporsi data pelatihan dan data pengujian, yaitu 70%-30%, 80%-20%, dan 90%-10%. Jumlah data pelatihan yang diberi label digunakan dengan tiga jenis ukuran, yaitu 10%, 20%, dan 30% dari jumlah keseluruhan data pelatihan. Uji coba algoritme *Self-Training Naive Bayes* menggunakan dimensi *Word2Vec* terbaik dari percobaan sebelumnya, yaitu 200. Parameter nilai batas ambang *confidence* (ε) yang digunakan algoritme *Self-Training* dalam percobaan ini adalah 0,7 sampai 0,9. Hasil percobaan ditampilkan menggunakan grafik pada Gbr. 4 sampai Gbr. 6.

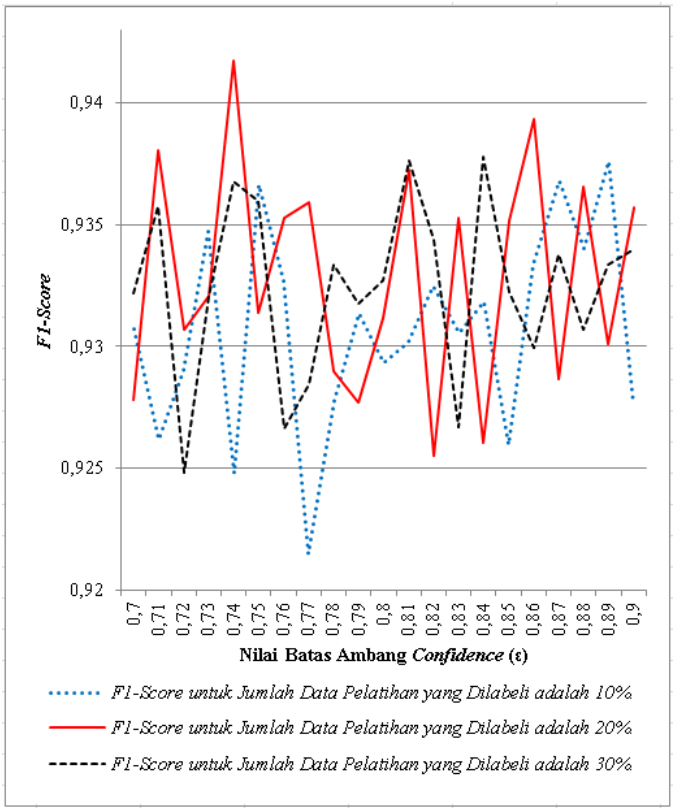


Gbr. 5 Grafik nilai *F1-Score* untuk algoritme *Self-Training Naive Bayes* dengan *percentage-splitting* 80%-20%.

Gbr. 4 merupakan grafik hasil pengujian dengan *percentage-splitting* 70%-30%. Hasil tersebut menunjukkan nilai *F1-Score* terbaik diperoleh dengan pengaturan nilai batas ambang *confidence* adalah 0,72 dengan jumlah data pelatihan yang dilabeli adalah 30%. Gbr. 5 merupakan grafik hasil pengujian dengan *percentage-splitting* 80%-20%. Hasil terbaik diperoleh dari percobaan dengan nilai batas ambang *confidence* adalah 0,71 dengan jumlah data pelatihan yang dilabeli adalah 30%. Hasil percobaan pada Gbr. 6 adalah hasil *percentage-splitting* 90%-10%. Hasil percobaan tertinggi diperoleh dengan nilai batas ambang *confidence* adalah 0,74 dengan jumlah data pelatihan yang dilabeli adalah 20%. Nilai *F1-Score* terbaik yang diperoleh adalah 0,9417 dengan pengujian *percentage-splitting* 90%-10% dan nilai batas ambang *confidence* sebesar 0,74. Jumlah data pelatihan yang digunakan dalam hasil tersebut adalah 5.330 data pelatihan yang diberi label dan 21.299 data pelatihan yang tak berlabel.

Dari ketiga grafik yang disajikan dapat disimpulkan bahwa kinerja terbaik diperoleh pada distribusi data pelatihan

berlabel dengan jumlah 20% dari total data pelatihan. Hal ini dapat dilihat dari garis lurus di Gbr. 4 sampai Gbr. 6 yang memiliki rata-rata kinerja lebih baik jika dibandingkan dengan garis lainnya.



Gbr. 6 Grafik nilai *F1-Score* untuk algoritme *Self-Training Naive Bayes* dengan *percentage-splitting* 90%-10%.

C. Evaluasi Perbandingan Kinerja Algoritme Self-Training Naive Bayes dengan Penelitian Sebelumnya

Pada bagian ini dilakukan evaluasi perbandingan kinerja algoritme *Self-Training Naive Bayes* dengan algoritme yang telah digunakan pada penelitian sebelumnya. Perbandingan kinerja yang pertama dilakukan dengan penelitian klasifikasi artikel Wikipedia menggunakan pendekatan *supervised learning* [6]. Penelitian tersebut merupakan bagian dari *preprocessing* dari pembangunan sebuah *ontology* bahasa Indonesia. Algoritme yang digunakan adalah *Multinomial Naive Bayes* yang dimodelkan secara pararel dengan *Map Reduce* menggunakan fitur *bag-of-words*.

Uji coba dilakukan dua kali untuk membandingkan kinerja usulan metode dengan algoritme dari penelitian sebelumnya. Dimensi *Word2Vec* yang digunakan sebagai fitur pada percobaan yang dilakukan adalah 200. Data yang digunakan dalam uji coba pertama berasal dari penelitian sebelumnya [6]. Data pelatihan yang digunakan berjumlah 2.000 artikel yang terdiri atas dua topik, yaitu hewan dan tumbuhan.

Pengujian dilakukan dengan ukuran data pengujian sesuai dengan penelitian sebelumnya, yaitu 200, 500, dan 1.000 artikel. Metode evaluasi yang digunakan untuk percobaan pertama adalah akurasi. Hasil kinerja dari percobaan pertama ini disajikan pada Tabel VI.

TABEL VI
HASIL PERBANDINGAN KINERJA *SELF-TRAINING NAIVE BAYES* DENGAN *MULTINOMIAL NAIVE BAYES* DI PENELITIAN SEBELUMNYA MENGGUNAKAN DATA WIKIPEDIA BAHASA INDONESIA

No.	Jumlah Data Pengujian	*Multinomial Naive Bayes*	*Self-Training Naive Bayes*
1	200 artikel	98,4 %	97,2% (ε = 0,8, jumlah data berlabel = 30%)
2	500 artikel	98,8 %	97,8% (ε = 0.8, jumlah data berlabel = 30%)
3	1.000 artikel	98,1%	96,1% (ε = 0,78, jumlah data berlabel = 20%)

Percobaan *Self-Training Naive Bayes* pada data pengujian berjumlah 200 artikel memperoleh nilai akurasi terbaik 97,2% dengan parameter batas ambang *confidence* (ε) adalah 0,8. Data latih yang diberi label berjumlah 30% dari total keseluruhan data pelatihan.

Self-Training Naive Bayes mendapatkan akurasi 97,8% pada data pengujian dengan jumlah 500 artikel. Hasil ini diperoleh dengan parameter batas ambang *confidence* (ε) adalah 0,8. Jumlah data yang diberi label adalah 30% dari data pelatihan yang digunakan.

Pada percobaan dengan jumlah 1.000 artikel, *Self-Training Naive Bayes* mampu memperoleh nilai akurasi 96,1%. Parameter batas ambang *confidence* (ε) yang digunakan adalah 0,8 dengan data yang diberi label berjumlah 20% dari keseluruhan data pelatihan. Pendekatan *semi-supervised learning* pada makalah ini dapat memberikan kinerja dengan beda rata-rata nilai akurasi 1,4% jika dibandingkan dengan hasil dari algoritme *supervised learning* dari penelitian sebelumnya.

Percobaan kedua dilakukan dengan pengujian *percentage-splitting* dengan pembagian 70-30 dan 80-20. Data yang digunakan pada percobaan kedua ini adalah berita bahasa Indonesia di penelitian dalam makalah ini. Evaluasi dilakukan dengan menggunakan nilai *F1-Score*. Hasil dari percobaan kedua diperlihatkan pada Tabel VII.

Kinerja dari algoritme *Multinomial Naive Bayes* di percobaan kedua memberikan nilai *F1-Score* 98,18% untuk pengujian *percentage-splitting* 70-30. Nilai *F1-Score* terbaik untuk algoritme *Self-Training Naive Bayes* pada pengujian *percentage-splitting* mencapai 93,79%. Parameter batas ambang *confidence* (ε) terbaik yang diperoleh adalah 0,76. Data yang diberi label secara manual pada percobaan ini berjumlah 20% dari total data pelatihan.

Algoritme *Multinomial Naive Bayes* di percobaan kedua untuk pengujian *percentage-splitting* 80-20 menghasilkan nilai *F1-Score* 98,29%. Algoritme *Self-Training Naive Bayes* yang ditawarkan pada pengujian *percentage-splitting* 80-20 di percobaan kedua memberikan nilai *F1-Score* 93,96%. Parameter batas ambang *confidence* (ε) terbaik yang

digunakan dalam percobaan adalah 0,71. Data yang diberi label secara manual di percobaan ini berjumlah 30% dari keseluruhan total data pelatihan. Beda rata-rata hasil nilai *F1-Score* dari pendekatan *semi-supervised learning* dan *supervised learning* pada percobaan ini adalah 6,5 %.

TABEL VII
HASIL PERBANDINGAN KINERJA *SELF-TRAINING NAIVE BAYES* DENGAN *MULTINOMIAL NAIVE BAYES* DI PENELITIAN SEBELUMNYA MENGGUNAKAN DATA BERITA BAHASA INDONESIA

No.	*Percentage-Splitting*	*Multinomial Naive Bayes*	*Self-Training Naive Bayes*
		F1-Score	***F1-Score***
1	70-30 (Data Pelatihan : 20.711dan Data Pengujian : 8.876)	98,18 %	93,79% (ε = 0,76, jumlah data berlabel = 20%)
2	80-20 (Data Pelatihan : 23.670 dan Data Pengujian : 5.917)	98,29 %	93,96% (ε = 0,71, jumlah data berlabel = 30%)

Perbandingan kinerja yang kedua dilakukan dengan klasifikasi berita *online* bahasa Indonesia pada penelitian sebelumnya [21]. Data yang digunakan dalam percobaan tersebut berjumlah 5.000 berita dari CNN Indonesia. Hasil kinerja terbaik di penelitian tersebut didapatkan dari kombinasi TF-IDF dan algoritme *Multinomial Naive Bayes*. Kinerja terbaik diperoleh dengan nilai *Precision* 98,41519%, nilai *Recall* 98,4%, dan nilai *F1-Score* diperoleh 98,407%.

Percobaan algoritme *Self-Training Naive Bayes* yang diusulkan menggunakan jumlah data hampir enam kali lebih besar jika dibandingkan jumlah data dari percobaan sebelumnya [21]. Penggunaan jumlah data pelatihan yang diberi label secara manual di penelitian ini lebih sedikit dan hasil kinerja nilai *F1-Score* terbaik hanya berbeda 4,28% jika dibandingkan dengan penelitian sebelumnya.

V. KESIMPULAN

Penggunaan model *Word2Vec Skip-Gram* sebagai fitur dalam kategorisasi teks memberikan nilai rata-rata *F1-Score* terbaik adalah 94,312% dengan dimensi terbaik adalah 200. Hasil rata-rata nilai *F1-score* dari seluruh percobaan dengan ukuran/dimensi yang berbeda-beda berada di angka mendekati 94%. Dari hasil percobaan yang dilakukan, dapat disimpulkan perubahan ukuran/dimensi dari *Word2Vec* tidak berpengaruh kepada hasil klasifikasi secara signifikan.

Penentuan nilai batas ambang *confidence* juga menentukan kinerja dari algoritme *Self-Training Naive Bayes* yang diusulkan. Hasil terbaik dari algoritme *Self-Training Naive Bayes* diperoleh dengan nilai *F1-Score* 0,9417 atau 94,17%. Hasil ini diperoleh dari pengujian *percentage-splitting* 90%-10% dengan parameter nilai batas ambang *confidence* (ε) adalah 0,74. Data yang diberi label hanya berjumlah 10% dari keseluruhan data latih. Secara umum, data diberi label sejumlah 20% dari total data pelatihan memiliki kinerja lebih baik dibandingkan dengan distribusi jumlah data berlabel lainnya. Hal ini dibuktikan dengan kinerja yang digambarkan oleh garis lurus pada Gbr. 4 sampai Gbr. 6.

Algoritme *Self-Training Naive Bayes* menggunakan data pelatihan yang telah dilabeli oleh pakar dengan jumlah lebih sedikit mampu mendekati kinerja dari algoritme *supervised learning* pada penelitian sebelumnya. Perbandingan dilakukan dengan dua penelitian sebelumnya. Hasil perbandingan dengan penelitian sebelumnya di Tabel VI memiliki selisih rata-rata sebesar 1,4%. Hasil perbadingan di Tabel VII memiliki selisih rata-rata sebesar 6,5% jika dibandingkan dengan penelitian yang dilakukan sebelumnya. Perbandingan kinerja terbaik dengan penelitian sebelumnya memberikan selisih sebesar 4,28% dengan jumlah data yang digunakan di makalah ini hampir enam kali lebih besar.

Pada penelitian mendatang, akan dicoba untuk menerapkan algoritme *semi-supervised learning* ini pada algoritme *Deep Learning* untuk melakukan klasifikasi teks. Upaya lain yang dilakukan dalam penelitian mendatang ialah penggunaan *word vector* selain *Word2Vec*, antara lain *Glove* dan *FastText* sebagai fitur klasifikasi.

UCAPAN TERIMA KASIH

Ucapan terima kasih diberikan kepada teman-teman mahasiswa di Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember atas bantuan dan saran-saran yang diberikan dalam menyelesaikan makalah ini. Terima kasih juga dihaturkan kepada mahasiswa Sekolah Tinggi Teknik Surabaya dalam bantuannya untuk mempersiapkan data yang digunakan.

REFERENSI

[1] (2017) Internet User Penetration in Indonesia from 2015 to 2022. [Online], https://www.statista.com/statistics/254460/internet-penetration-rate-in-indonesia/, tanggal akses : 01 Januari 2017.

[2] D. Rahmawati dan M. L. Khodra, "Word2vec Semantic Representation in Multilabel Classification for Indonesian News Article," *4th IGNITE Conf. 2016 Int. Conf. Adv. Informatics Concepts, Theory Appication. ICAICTA 2016*, 2016, hal. 1–6.

[3] W. Xu, H. Sun, C. Deng, dan Y. Tan, "Variational Autoencoder for Semi-Supervised Text Classification," *Proc. of the Thirty-First AAAI Conf. on Artificial Intelligence (AAAI-17)*, 2017, hal. 3358–3364.

[4] Z. Xu, J. Li, B. Liu, J. Bi, R. Li, dan R. Mao, "Semi-Supervised Learning in Large Scale Text Categorization," *Journal Shanghai Jiaotong University*, Vol. 22, No. 3, hal. 291–302, 2017.

[5] C. Olivier, B. Schölkopf, dan A. Zien, *Semi-Supervised Learning*, Cambridge, USA: The MIT Press, 2006.

[6] J. Santoso, E. M. Yuniarno, dan M. Hariadi, "Large Scale Text Classification using Map Reduce and Naive Bayes Algorithm for Domain Specified Ontology Building," *Proceedings - 2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2015*, 2015, Vol. 1, hal. 428–432.

[7] J. Lilleberg, Y. Zhu, dan Y. Zhang, "Support Vector Machines and Word2vec for Text Classification with Semantic Features," *Proc. IEEE 14th International. Conference on Cognitive Informatics & Cognitive Computing*, 2015, hal. 136–140.

[8] Z. Gong dan T. Yu, "Chinese Web Text Classification System Model Based on Naive Bayes," *2010 International Conference on E-Product E-Service and E-Entertainment (ICEEE)*, 2010, hal. 1–4.

[9] A. Tripathy, A. Agrawal, dan S. K. Rath, "Classification of Sentimental Reviews Using Machine Learning Techniques," *Procedia Computer Science*, Vol. 57, hal. 821–829, 2015.

[10] Arifin dan K. E. Purnama, "Classification of Emotions in Indonesian TextsUsing K-NN Method," *International Journal of Information and Electronics Engineering*, Vol. 2, No. 3, hal. 899-903, Nov. 2012.

[11] A. Zaini, M. A. Muslim, dan Wijono, "Pengelompokan Artikel Berbahasa Indonesia Berdasarkan Struktur Laten Menggunakan Pendekatan Self Organizing Map," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, Vol. 6, No. 3, hal. 259-267, 2017.

[12] A. Blum dan T. Mitchell, "Combining Labeled and Unlabeled Data with Co-Training," *Proceedings of the Eleventh Annual Conference on Computational Learning Theory - COLT' 98*, 1998, hal. 92–100.

[13] S. Kiritchenko dan S. Matwin, "Email Classification with Co-Training," *Proceedings of the 2001 Conference of the Centre for Advanced Studies on Collaborative Research*, 2001, hal. 301–312.

[14] R. Guzmán-Cabrera, M. Montes-Y-Gómez, P. Rosso, dan L. Villaseñor-Pineda, "Using the Web as Corpus for Self-Training Text Categorization," *Information Retrieval*, Vol. 12, No. 3, hal. 400–415, 2009.

[15] J. Laksana dan A. Purwarianti, "Indonesian Twitter Text Authority Classification for Government in Bandung," *2014 International Conference of Advanced Informatics: Concept, Theory and Application (ICAICTA)*, 2014, hal. 129–134.

[16] A. Rachmania, J. Jaafar, dan N. Zamin, "Likelihood Calculation Classification for Indonesian Language News Documents," *2013 International Conference on Information Technology and Electrical Engineering (ICITEE)*, 2013, hal. 149–154.

[17] B. Y. Pratama dan R. Sarno, "Personality Classification Based on Twitter Text using Naive Bayes, KNN and SVM," *Proc. 2015 Int. Conf. Data Softw. Eng. ICODSE 2015*, 2015, hal. 170–174.

[18] A. R. Naradhipa dan A. Purwarianti, "Sentiment Classification for Indonesian Message in Social Media," *Proceedings of the 2011 International Conference on Electrical Engineering and Informatics*, 2011, hal. 2–5.

[19] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, dan J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," *Advances in Neural Information Processing Systems*, 2013, hal. 3111–3119.

[20] C. Xing, D. Wang, X. Zhang, and C. Liu, "Document Classification with Distributions of Word Vectors," *Asia-Pacific Signal and Information Processing Association, 2014 Annual Summit and Conference (APSIPA)*, 2014, hal. 1–5.

[21] R. Wongso, F. A. Luwinda, B. C. Trisnajaya, O. Rusli, and Rudy, "News Article Text Classification in Indonesian Language," *Procedia Computer Science*, Vol. 116, hal. 137–143, 2017.

[22] F. Z. Tala, "A Study of Stemming Effects on Information Retrieval in Bahasa Indonesia," M.Sc. thesis, University of Amsterdam, Netherlands, 2003.

[23] O. Somantri dan M. Khambali, "Feature Selection Klasifikasi Kategori Cerita Pendek Menggunakan Naive Bayes dan Algoritme Genetika," *Jurnal Nasional Teknik Elektro dan Teknologi Informasi*, Vol. 6, No. 3, hal. 301-306, 2017.

[24] M. Naili, A. H. Chaibi, dan H. H. Ben Ghezala, "Comparative Study of Word Embedding Methods in Topic Segmentation," *Procedia Computer Science*, Vol. 112, hal. 340–349, 2017.

[25] A. Søgaard, "Semi-Supervised Learning and Domain Adaptation in Natural Language Processing," *Synth. Lect. Hum. Lang. Technol.*, Vol. 6, No. 2, hal. 1–103, 2013.

[26] A. Fujino, H. Isozaki, dan J. Suzuki, "Multi-label Text Categorization with Model Combination based on F1-score Maximization," *Proc. IJCNLP*, 2008, hal. 823–828.