

Terbit online pada laman : <http://teknosi.fti.unand.ac.id/>

Jurnal Nasional Teknologi dan Sistem Informasi

| ISSN (Print) 2460-3465 | ISSN (Online) 2476-8812 |



Artikel Penelitian

Analisis Sentimen *Twitter* untuk Menilai Opini Terhadap Perusahaan Publik Menggunakan Algoritma *Deep Neural Network*

Erwin Yudi Hidayat^{a,*}, Raindy Wicaksana Hardiansyah^a, Affandy^a^a Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Semarang, 50131, Indonesia

INFORMASI ARTIKEL

Sejarah Artikel:

Diterima Redaksi: 13 Juli 2021

Revisi Akhir: 06 September 2021

Diterbitkan Online: 10 September 2021

KATA KUNCI

Analisis Sentimen,
Twitter,
Klasifikasi,
Perusahaan Publik,
Deep Neural Network

KORESPONDENSI

E-mail: erwin@dsn.dinus.ac.id*

ABSTRACT

Dalam menaikkan kinerja serta mengevaluasi kualitas, perusahaan publik membutuhkan feedback dari masyarakat / konsumen yang bisa didapat melalui media sosial. Sebagai pengguna media sosial *Twitter* terbesar ketiga di dunia, tweet yang beredar di Indonesia memiliki potensi meningkatkan reputasi dan citra perusahaan. Dengan memanfaatkan algoritma *Deep Neural Network* (DNN), *neural network* yang tersusun dari *layer* yang jumlahnya lebih dari satu, didapati hasil analisa sentimen pada *Twitter* berbahasa Indonesia menjadi lebih baik dibanding dengan metode lainnya. Penelitian ini menganalisa sentimen melalui *tweet* dari masyarakat Indonesia terhadap sejumlah perusahaan publik dengan menggunakan DNN. Data *Tweet* sebanyak 5504 record didapat dengan melakukan *crawling* melalui *Application Programming Interface* (API) *Twitter* yang selanjutnya dilakukan *preprocessing* (*cleansing*, *case folding*, formalisasi, *stemming*, dan tokenisasi). Proses labeling dilakukan untuk 3902 record dengan memanfaatkan aplikasi *Sentiment Strength Detection*. Tahap pelatihan model dilakukan menggunakan algoritma DNN dengan variasi jumlah *hidden layer*, susunan node, dan nilai *learning rate*. Eksperimen dengan proporsi data training dan testing sebesar 90:10 memberikan hasil performa terbaik. Model tersusun dengan 3 *hidden layer* dengan susunan *node* tiap *layer* pada model tersebut yaitu 128, 256, 128 *node* dan menggunakan *learning rate* sebesar 0.005, model mampu menghasilkan nilai akurasi mencapai 88.72%.

1. PENDAHULUAN

Perusahaan Publik adalah Perseroan Terbatas seperti yang dimaksud dalam Pasal 1 angka 1 Ketentuan Umum Undang-undang Nomor 40 Tahun 2007 tentang Perseroan Terbatas. Sahamnya telah dimiliki sekurang-kurangnya oleh 300 (tiga ratus) pemegang saham dan memiliki modal disetor sekurang-kurangnya Rp 3.000.000.000 (tiga miliar rupiah) atau suatu jumlah pemegang saham dan modal disetor yang ditetapkan dengan Peraturan Pemerintah [1]. Dalam menaikkan kinerja serta mengevaluasi kualitas dalam suatu perusahaan maka dibutuhkan *feedback* berupa opini yang dapat didapatkan dari *website* perusahaan tersebut ataupun melalui sosial media [2].

Twitter adalah *microblogging platform* yang sangat populer. Layanan jejaring sosial ini memungkinkan pengguna untuk membagikan 240 karakter melalui suatu *tweet* [3]. Pada pertengahan tahun 2015, jumlah pengguna *Twitter* di Indonesia mencapai lima puluh juta pengguna. Hal ini menjadikan Indonesia sebagai negara dengan pengguna aktif *Twitter* terbanyak ketiga di dunia [4]. Dengan pengelolaan yang tepat, *Twitter* dapat dimanfaatkan sebagai ladang keuntungan bagi perusahaan atau seseorang yang memiliki kepentingan-kepentingan tertentu untuk meningkatkan reputasi dan citranya. Media tersebut banyak digunakan oleh masyarakat untuk sekedar meluapkan emosinya maupun membagikan suatu informasi [3]. Banyak perusahaan publik yang memiliki akun *Twitter* untuk memberikan informasi dan berinteraksi dengan konsumennya. Interaksi tersebut dilakukan melalui unggahan *Twitter* yang disebut *tweet*. Interaksi antar masyarakat dan perusahaan

tentunya dapat berupa pendapat, saran, ataupun kritik terhadap suatu hal [5]. *Tweet* tentang perusahaan publik sangat beragam. Masyarakat dapat menulis *tweet* yang berupa komentar positif ataupun negatif tentang suatu perusahaan. Perusahaan publik dapat memanfaatkan hal tersebut untuk mengetahui bagaimana citra perusahaan di masyarakat. Namun terjadi kesulitan dalam pemanfaatan itu pada saat melakukan klasifikasi kalimat, mengingat banyaknya unggahan *tweet* maka dibutuhkan waktu dan usaha yang besar jika proses klasifikasi dilakukan secara manual [4]. Oleh karena itu perlu dilakukan analisis sentimen.

Analisis sentimen merupakan bagian dari *text mining* yang mempelajari komputasi opini orang-orang, emosi, dan sentimen dan bertujuan untuk mengklasifikasikan sentimen pada suatu kalimat yang berupa opini. Analisis sentimen dapat menentukan apakah kalimat tersebut merupakan kalimat positif atau negatif berdasarkan polaritasnya [6]. Analisa sentimen dapat digunakan untuk mengetahui sentimen pengguna *Twitter* terhadap perusahaan publik. *Tweet* dapat dikategorikan berdasarkan sentimen kalimatnya dan dapat memberikan informasi terhadap tanggapan masyarakat yang memengaruhi citra perusahaan publik. Penelitian mengenai analisis sentimen dilakukan oleh [7] yang membahas klasifikasi sentimen wisatawan Candi Borobudur pada situs *Tripadvisor* menggunakan *Support Vector Machine* (SVM) dan *K-Nearest Neighbour*. Kemudian penelitian lain juga dilakukan oleh [8] membahas mengenai analisis sentimen terhadap ulasan hotel menggunakan *Boosting Weighted Extreme Learning Machine*. Lalu [9] melakukan penelitian tentang analisis sentimen berbasis aspek pada ulasan pelanggan restoran Bakso President Malang dengan metode *Naïve Bayes Classifier*. Dalam penelitian tersebut data set dilakukan *preprocessing* terlebih dahulu sehingga data yang akan diklasifikasi menjadi bersih dan siap untuk diolah menggunakan algoritma tertentu.

Artificial Neural Network (ANN) adalah sistem komputasi yang diadopsi dari jaringan syaraf otak manusia yang memiliki kemampuan untuk belajar dan berkembang. Pada ANN, Jaringan syaraf ini diinterpretasikan menjadi graf yang memiliki bobot dan arah [10]. Salah satu penerapan dari *neural network* adalah *Deep Neural Network* atau DNN. *Deep Neural Network* adalah *neural network* yang tersusun dari *layer* yang jumlahnya lebih dari satu [11]. Penelitian yang menggunakan DNN dilakukan oleh [12] untuk menganalisa sentimen pada *Twitter* berbahasa Indonesia mengenai institusi pemerintahan dan tokoh pemerintahan. Penelitian tersebut membandingkan metode *Deep Learning Neural Network* dengan SVM dan *Naïve Bayes*. Hasil penelitian yang didapat menunjukkan bahwa performa metode DNN lebih baik dari metode lainnya.

Penelitian ini bertujuan menerapkan metode DNN untuk melakukan analisis sentimen pada *tweet* berbahasa Indonesia mengenai perusahaan publik di Indonesia. Akurasi dari DNN juga menjadi bahasan yang diangkat. Dari tujuan tersebut, penelitian dapat mendukung perusahaan dalam melakukan analisis sentimen opini publik terhadap reputasi atau citra perusahaan publik yang ada di masyarakat. Sehingga, perusahaan dapat mendapatkan umpan balik untuk menentukan langkah strategis dan kebijakan yang akan ditempuh selanjutnya.

2. METODE

2.1 Sumber Data

Penelitian ini menggunakan sumber data yang berupa *tweet* yang diambil secara langsung dari sosial media *Twitter*. Data diambil dengan memanfaatkan *Application Programming Interface* (API) yang telah disediakan oleh *Twitter*.

Data yang diambil pada penelitian ini menggunakan beberapa kata kunci seperti 'myxlcare', 'telkomsel', 'indihome', 'satpam bca', 'indomie', 'indofood'. Kata kunci tersebut merupakan kata kunci pada *tweet* yang berkaitan dengan perusahaan publik di Indonesia, seperti Unilever, Indofood, BCA, BRI, Maybank, Telkom Indonesia, dan XL Axiata. Data tersebut kemudian disimpan pada file yang berformat *Comma Separated Values* (CSV).

2.2 Metode Pengumpulan Data

Studi literatur dilakukan dengan cara mengumpulkan data dan informasi melalui buku, literatur ilmiah, jurnal, dan media online yang dapat dijadikan sebagai pendukung untuk menunjang referensi dan landasan teori yang sesuai dengan penelitian ini.

Observasi dilakukan dengan cara mengamati dan mengambil data secara langsung pada sosial media *Twitter*. Pengambilan dilakukan dengan memanfaatkan *Twitter* API yang telah disediakan oleh *Twitter*.

2.3 Tahap Penelitian

Data diambil dengan memanfaatkan API yang telah disediakan oleh *Twitter*. Untuk menggunakan *Twitter* API, sebelumnya harus melakukan pendaftaran akun pengembang. Akun tersebut dapat digunakan untuk mendapatkan API *key* dan *access token* yang berperan sebagai kunci pengaksesan *Twitter* API. Kemudian digunakan *library* Tweepy untuk mempermudah pengaksesan *Twitter* API pada Python. Teknik *crawling* yang digunakan adalah teknik dengan menggunakan kata kunci nama perusahaan atau produk dari suatu perusahaan.

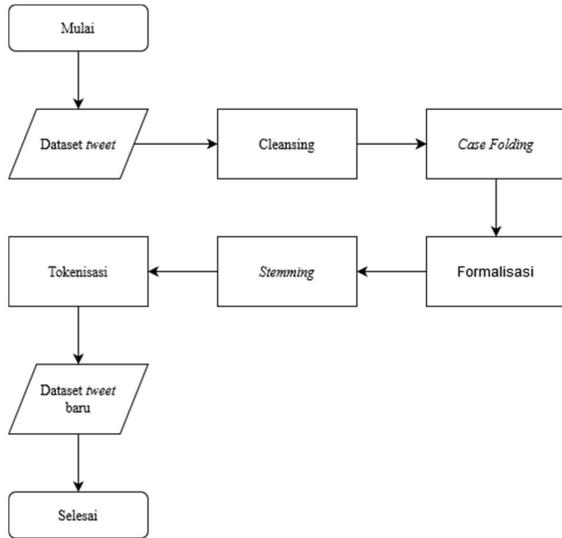
Data set yang sudah terambil kemudian dilakukan *preprocessing* yang bertujuan untuk mengurangi *noise* sehingga data yang dihasilkan menjadi terstruktur dan siap untuk diolah menggunakan metode DNN. Tahapan yang ada pada *preprocessing* ini antara lain *case folding* yaitu proses dimana seluruh karakter pada data set diubah menjadi huruf kecil. Kemudian *cleansing* yaitu proses untuk menghapus URL, *mention*, *hashtag*, *retweet*, *symbol*, dan angka. Lalu dilakukan perubahan kata pada data set menjadi kata baku. Setelah itu dilakukan *stemming* untuk mengubah kata pada data set menjadi kata dasar. Data tersebut kemudian diterapkan tokenisasi untuk memisahkan kata pada setiap teks menjadi kata yang independen

Dataset yang sudah diproses dibagi menjadi dua yaitu data *training* dan data *testing*. Data *training* digunakan sebagai data untuk melatih model sedangkan data *testing* digunakan sebagai data untuk melakukan pengujian model.

Data set kemudian diklasifikasikan dengan metode DNN dan dilakukan pengujian untuk mengukur performa model.

2.4 Text Processing

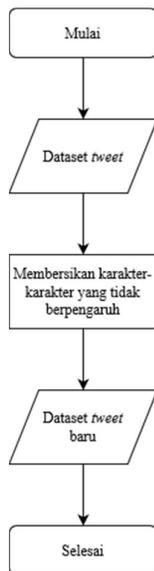
Text Preprocessing merupakan proses pengolahan teks yang bertujuan untuk mengurangi *noise* pada data set serta mengubah data set menjadi bentuk yang lebih terstruktur. Tahapan *text preprocessing* yang diterapkan pada penelitian ini adalah *cleansing*, *case folding*, formalisasi, *stemming*, dan tokenisasi. Berikut merupakan tahapan *text preprocessing* yang ditunjukkan pada Gambar 1.



Gambar 1. Flowchart Text Preprocessing

2.4.1 Cleansing

Proses *cleansing* merupakan proses dimana teks dilakukan penyederhanaan terhadap karakter atau komponen yang tidak berpengaruh untuk proses klasifikasi. Proses ini menghapus komponen data seperti duplikasi, *mention*, atribut yang diawali dengan *retweet*, dan atribut yang diawali dengan *hashtag*, URL, simbol, dan angka. Gambar 2 merupakan *flowchart* dari *cleansing*, dan contoh *cleansing* ditunjukkan oleh Tabel 1.



Gambar 2. Flowchart Cleansing

Tabel 1. Contoh *Cleansing*

| Masukan | Keluaran |
|--|--|
| Emang ya pegawai paling ramah cuma satpam bank bca, bank sulsebar sama pramuniaga uniqlo. yang lain jutek bgt padahal kan ya dia digaji dari hasil belanjaan gue | Emang ya pegawai paling ramah cuma satpam bank bca bank sulsebar sama pramuniaga uniqlo yang lain jutek bgt padahal kan ya dia digaji dari hasil belanjaan gue |
| @IchaVarma satpam bank bca emg terbaik sih pd sopan dan ramah bgt | satpam bank bca emg terbaik sih pd sopan dan ramah bgt |
| Sekarang indihome terlalu sering gangguan gk asik banget indihome skrng gk pernah berbenah kalah ama tetangganya. | Sekarang indihome terlalu sering gangguan gk asik banget indihome skrng gk pernah berbenah kalah ama tetangganya |
| Astagfirullah malam malam pengen bikin indomie, kalau aku bikin sangat berdosa sekali pada tubuh ini :)) | Astagfirullah malam malam pengen bikin indomie kalau aku bikin sangat berdosa sekali pada tubuh ini |
| @safirahtasa Makanan aja sih gue banyak maunya parah. Ditambah psbb makin banyak maunya dan jauh waktu itu gue mimpi makan es krim walls strawberry cheesecake di pabriknya jd sekeliling gue es krim semua. Pas bangun pengen bgt tapi jam 2 pagi dinihari :) | Makanan aja sih gue banyak maunya parah Ditambah psbb makin banyak maunya dan jauh waktu itu gue mimpi makan es krim walls strawberry cheesecake di pabriknya jd sekeliling gue es krim semua Pas bangun pengen bgt tapi jam pagi dinihari |

2.4.2 Case Folding

Case folding merupakan proses untuk memeriksa semua karakter pada data set dan mengubahnya menjadi huruf kecil. Gambar 3 merupakan *flowchart* dari *case folding*.



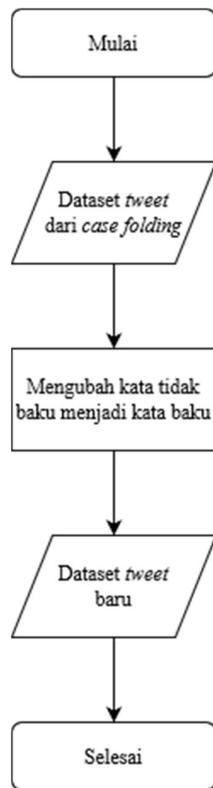
Gambar 3. Flowchart Case Folding

Tabel 2. Contoh *Case Folding*

| Masukan | Keluaran |
|--|--|
| Emang ya pegawai paling ramah cuma satpam bank bca bank suselbar sama pramuniaga uniqlo yang lain jutek bgt padahal kan ya dia digaji dari hasil belanjaan gue | emang ya pegawai paling ramah cuma satpam bank bca bank suselbar sama pramuniaga uniqlo yang lain jutek bgt padahal kan ya dia digaji dari hasil belanjaan gue |
| satpam bank bca emg terbaik sih pd sopan dan ramah bgt | satpam bank bca emg terbaik sih pd sopan dan ramah bgt |
| Sekarang indihome terlalu sering gangguan gk asik banget indihome skrng gk pernah berbenah kalah ama tetangganya | sekarang indihome terlalu sering gangguan gk asik banget indihome skrng gk pernah berbenah kalah ama tetangganya |
| Astagfirullah malam malam pengen bikin indomie kalau aku bikin sangat berdosa sekali pada tubuh ini | astagfirullah malam malam pengen bikin indomie kalau aku bikin sangat berdosa sekali pada tubuh ini |
| Makanan aja sih gue banyak maunya parah Ditambah psbb makin banyak maunya dan jauh Ter random waktu itu gue mimpi makan es krim walls strawbery cheesecake di pabriknya jd sekeliling gue es krim semua Pas bangun pengen bgt tapi jam pagi dinihari | makanan aja sih gue banyak maunya parah ditambah psbb makin banyak maunya dan jauh ter random waktu itu gue mimpi makan es krim walls strawbery cheesecake di pabriknya jd sekeliling gue es krim semua pas bangun pengen bgt tapi jam pagi dinihari |

2.4.3 *Formalisasi*

Formalisasi merupakan proses pengubahan setiap kata yang memiliki ejaan yang tidak tepat menjadi kata yang sesuai dengan kaidah Bahasa Indonesia. Gambar 4 merupakan *flowchart* dari formalisasi. Tabel 3 merupakan contoh dari formalisasi.



Gambar 4. *Flowchart* Formalisasi

Tabel 3. Contoh Formalisasi

| Masukan | Keluaran |
|--|--|
| emang ya pegawai paling ramah cuma satpam bank bca bank suselbar sama pramuniaga uniqlo yang lain jutek bgt padahal kan ya dia digaji dari hasil belanjaan gue | memang iya pegawai paling ramah cuma satuan pengamanan bank bca bank suselbar sama pramuniaga uniqlo yang lain galak banget padahal kan iya dia digaji dari hasil belanjaan saya |
| satpam bank bca emg terbaik sih pd sopan dan ramah bgt | satpam bank bca memang terbaik sih pada sopan dan ramah banget |
| sekarang indihome terlalu sering gangguan gk asik banget indihome skrng gk pernah berbenah kalah ama tetangganya | sekarang indihome terlalu sering gangguan tidak asik banget indihome sekarang tidak pernah berbenah kalah sama tetangganya |
| astagfirullah malam malam pengen bikin indomie kalau aku bikin sangat berdosa sekali pada tubuh ini | astagfirullah malam malam ingin bikin indomie kalau aku bikin sangat berdosa sekali pada tubuh ini |
| makanan aja sih gue banyak maunya parah ditambah psbb makin banyak maunya dan jauh Ter random waktu itu gue mimpi makan es krim walls strawbery cheesecake di pabriknya jd sekeliling gue es krim semua Pas bangun pengen bgt tapi jam pagi dinihari | makanan saja sih saya banyak maunya parah ditambah psbb makin banyak maunya dan jauh ter random waktu itu saya mimpi makan es kirim walls strawbery cheesecake di pabriknya jadi sekeliling saya es kirim semua pas bangun ingin banget tapi jam pagi dinihari |

2.4.4 *Stemming*

Stemming merupakan proses untuk mengubah kata yang memiliki imbuhan menjadi kata akar atau kata dasarnya. Proses *stemming* pada penelitian ini dilakukan dengan menggunakan *library* Sastrawi. Gambar 5 merepresentasikan *flowchart* bagaimana *stemming* dikerjakan.



Gambar 5. *Flowchart* *Stemming*

Adaupun Tabel 4 menggambarkan contoh dari *stemming*.

Tabel 4. Contoh *Stemming*

| Masukan | Keluaran |
|--|---|
| memang iya pegawai paling ramah cuma satuan pengamanan bank bca bank sulselbar sama pramuniaga uniqlo yang lain galak banget padahal kan iya dia digaji dari hasil belanjaan saya | memang iya pegawai paling ramah cuma satu aman bank bca bank sulselbar sama pramuniaga uniqlo yang lain galak banget padahal kan iya dia gaji dari hasil belanja saya |
| satuan pengamanan bank bca memang terbaik sih pada sopan dan ramah banget | satpam bank bca memang baik sih pada sopan dan ramah banget |
| sekarang indihome terlalu sering gangguan tidak asik banget indihome sekarang tidak pernah berbenah kalah sama tetangganya | sekarang indihome terlalu sering ganggu tidak asik banget indihome sekarang tidak pernah benah kalah sama tetangga |
| astagfirullah malam malam ingin bikin indomie kalau aku bikin sangat berdosa sekali pada tubuh ini | astagfirullah malam malam ingin bikin indomie kalau aku bikin sangat dosa sekali pada tubuh ini |
| makanan saja sih saya banyak maunya parah ditambah psbb makin banyak maunya dan jauh ter random waktu itu saya mimpi makan es kirim walls strawbery cheesecake di pabriknya jadi sekeliling saya es kirim semua pas bangun ingin banget tapi jam pagi dinihari | makan saja sih saya banyak mau parah tambah psbb makin banyak mau dan jauh ter random waktu itu saya mimpi makan es kirim walls strawbery cheesecake di pabrik jadi keliling saya es kirim semua pas bangun ingin banget tapi jam pagi dinihari |

2.4.5 Tokenisasi

Tokenisasi adalah proses untuk memotong setiap kata pada data set menjadi kata yang independen. Gambar 6 merupakan *flowchart* dari tokenisasi, dengan contoh tokenisasi ditunjukkan pada Tabel 5.



Gambar 6. *Flowchart* Tokenisasi

Tabel 5. Contoh Tokenisasi

| Masukan | Keluaran |
|---|---|
| memang iya pegawai paling ramah cuma satu aman bank bca bank sulselbar sama pramuniaga uniqlo yang lain galak banget padahal kan iya dia gaji dari hasil belanja saya | {'memang' 'iya' 'pegawai' 'paling' 'ramah' 'cuma' 'satu' 'aman' 'bank' 'bca' 'bank' 'sulselbar' 'sama' 'pramuniaga' 'uniqlo' 'yang' 'lain' 'galak' 'banget' 'padahal' 'kan' 'iya' 'dia' 'gaji' 'dari' 'hasil' 'belanja' 'saya'} |
| satu aman bank bca memang baik sih pada sopan dan ramah banget | {'satpam' 'bank' 'bca' 'memang' 'baik' 'sih' 'pada' 'sopan' 'dan' 'ramah' 'banget'} |
| sekarang indihome terlalu sering gangguan tidak asik banget indihome sekarang tidak pernah benah kalah sama tetangga | {'sekarang' 'indihome' 'terlalu' 'sering' 'ganggu' 'tidak' 'asik' 'banget' 'indihome' 'sekarang' 'tidak' 'pernah' 'benah' 'kalah' 'sama' 'tetangga'} |
| astagfirullah malam malam ingin bikin indomie kalau aku bikin sangat dosa sekali pada tubuh ini | {'astagfirullah' 'malam' 'malam' 'ingin' 'bikin' 'indomie' 'kalau' 'aku' 'bikin' 'sangat' 'dosa' 'sekali' 'pada' 'tubuh' 'ini'} |
| makan saja sih saya banyak mau parah tambah psbb makin banyak mau dan jauh ter random waktu itu saya mimpi makan es kirim walls strawbery cheesecake di pabrik jadi keliling saya es kirim semua pas bangun ingin banget tapi jam pagi dinihari | {'makan' 'saja' 'sih' 'saya' 'banyak' 'mau' 'parah' 'tambah' 'psbb' 'makin' 'banyak' 'mau' 'dan' 'jauh' 'ter' 'random' 'waktu' 'itu' 'saya' 'mimpi' 'makan' 'es' 'kirim' 'walls' 'strawbery' 'cheesecake' 'di' 'pabrik' 'jadi' 'keliling' 'saya' 'es' 'kirim' 'semua' 'pas' 'bangun' 'ingin' 'banget' 'tapi' 'jam' 'pagi' 'dinihari'} |

2.5 Labeling

Proses *labeling* dilakukan menggunakan aplikasi Sentiment Strength Detection dari hasil penelitian yang dilakukan oleh [13]. Aplikasi tersebut merupakan aplikasi *labeling* berbasis kamus yang meliputi kamus *sentivord*, *questionword*, *negatingword*, *emoticon*, dan *boosterword*.

Tabel 6. Sampel *tweet* setelah dilakukan *labeling*

| No. | <i>Tweet</i> | Label |
|-----|--|---------|
| 1 | Satpam BCA emang terbaik sih, padahal cuma nganter temen tapi diperlakukan kayak putri raja uhhh manjah wkwk | Positif |
| 2 | Beberapa hari lalu update @myXL @myXLCare tapi belum sempet buka karena pikir 'ah paling update minor doang' dan pagi ini terkaget-kaget karena ternyata 'major' update dan aplikasinya bagus banget, jauh lebih smooth dibanding myXL versi sebelumnya. @XLAXiata_Tbk https://t.co/FkatRYp6r0 | Positif |
| 3 | Pantesan pulsa XL aing habis ga jelas trs. Taunya tiba-tiba terdaftar kaya ginian yg ga pernah aing ikutan. Termasuk penipuan konsumen ga sih? @myXLCare @myXL @TMCPoldaMetro https://t.co/lFRbNJT0xt https://t.co/co5Pp8XKeL;myxlcare | Negatif |
| 4 | @guiijjsee @myXLCare Tolong dong, daerah ponorogo juga. Dibuat wa bisa tapi untuk social media yang lain MasyaAllah lemot nya. | Negatif |
| 5 | @sptwnangga Menurut lo recommended yang mana? Sambal Tabur Boncabe atau Jawara cabai tabur? | Netral |
| 6 | @IndiHome malam, mau nanya kalo daerah bekasi di daerah caman sudah tercover? terima kasih | Netral |

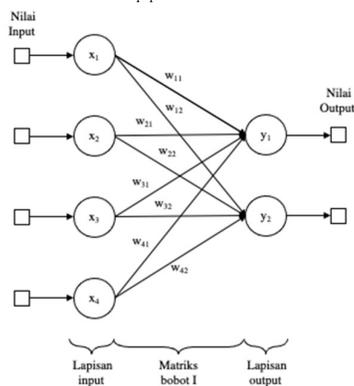
Untuk menguji ketepatan aplikasi tersebut maka diambil masing-masing 20 data positif, 20 data negatif, dan 20 data netral dari data set. Data tersebut sebelumnya telah diberi label secara manual dan kemudian diuji dengan menggunakan aplikasi Sentiment Strength Detection. Hasil dari pengujian didapatkan akurasi dari aplikasi *labeling* tersebut adalah 96.67%. Tabel 6 sampel dari *tweet* yang dilakukan proses *labeling*.

2.6 Artificial Neural Network

Artificial Neural Network adalah sistem komputasi yang diadopsi dari jaringan syaraf otak manusia yang memiliki kemampuan untuk belajar dan berkembang. Pada ANN, jaringan syaraf ini diinterpretasikan menjadi graf yang memiliki bobot dan arah [10].

Salah satu penerapan dari *neural network* adalah DNN. *Deep Neural Network* adalah *neural network* yang tersusun dari *layer* yang jumlahnya lebih dari satu. Pada setiap *layer* tersebut memiliki *neuron* yang jumlahnya bervariasi pada setiap *layer*. Kemudian setiap *layer* melakukan perhitungan jumlah bobot *w* dikali dengan input *x* yang ditambah dengan bias. Hasil perhitungan ini kemudian dihitung lagi dengan menggunakan fungsi aktivasi yang merupakan keluaran dari *layer* tersebut. Persamaan 1 merupakan perhitungan pada setiap *layer* [11].

$$y = \alpha \left(\sum_{i=1}^n w_i * x_i + b \right) \tag{1}$$



Gambar 7. Ilustrasi ANN

Fungsi aktivasi yang paling umum digunakan pada *neural network* meliputi *hyperbolic tangent* (tanh), *sigmoid*, dan *rectified linear* (ReLU). Berikut merupakan persamaan dari masing-masing fungsi aktivasi tersebut.

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2}$$

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \tag{3}$$

$$\text{ReLU}(x) = \max(0, x) \tag{4}$$

Pada *layer* terakhir fungsi aktivasi yang dapat digunakan sebagai keluaran *neuron* adalah *softmax*. Fungsi *softmax* mengubah vektor n-dimensi yang berisi nilai riil menjadi vektor n-dimensi yang nilainya dalam rentang 0 sampai 1. Berikut merupakan persamaan dari fungsi *softmax* [14].

$$\text{softmax}(x)_j = \frac{e^x}{\sum_{k=1}^K e^x}, \text{ for } j=1, \dots, K \tag{5}$$

2.7 Metode yang Diusulkan

Metode yang digunakan dalam penelitian ini menerapkan metode DNN. Klasifikasi dilakukan dengan dua tahap yaitu klasifikasi menggunakan *training set* yang bertujuan untuk melatih model dan klasifikasi menggunakan *test set* yang bertujuan untuk menguji model yang telah dilatih.

Tabel 7 merupakan *vocabulary* yang berisi seluruh kata yang ada pada data set yang kemudian diberi indeks. Pada indeks 0 terdapat token <OOV> yang bertujuan untuk mengisi kata yang kosong agar bentuk input *sequence* yang akan diklasifikasikan dengan model DNN berjumlah tetap atau sama.

Tabel 7. Contoh *vocabulary*

| Indeks | Token |
|--------|--------|
| 0 | <OOV> |
| 1 | Produk |
| 2 | Ini |
| 3 | Sangat |
| 4 | Bagus |
| 5 | Jelek |
| 0 | <OOV> |

Pada Tabel 8 jumlah elemen *sequence* yang digunakan adalah sebanyak 6 elemen.

Tabel 8. Contoh Representasi Data

| Tweet | Tweet Sequence | Kelas |
|-------------------------|----------------------|---------|
| Produk ini sangat bagus | [1, 2, 3, 4, 0, 0] | Positif |
| Produk ini jelek | [1, 2, 5, 0, 0, 0] | Negatif |

Untuk mengklasifikasikan teks, model DNN memanfaatkan *word embedding* yang mengubah setiap kata dalam *vocabulary* menjadi vektor yang berisi angka. Tujuannya adalah untuk mengenali fitur pada setiap kata. Tabel 9 merupakan contoh *word embedding* dengan menggunakan dimensi *embedding* yang berjumlah *n* dimensi.

Tabel 9. Contoh *Word Embedding*

| Indeks Kata | Nilai Vektor |
|-------------|---|
| 0 | [e ₀₁ , e ₀₂ , e ₀₃ , ..., e _{0n}] |
| 1 | [e ₁₁ , e ₁₂ , e ₁₃ , ..., e _{1n}] |
| 2 | [e ₂₁ , e ₂₂ , e ₂₃ , ..., e _{2n}] |
| 3 | [e ₃₁ , e ₃₂ , e ₃₃ , ..., e _{3n}] |
| 4 | [e ₄₁ , e ₄₂ , e ₄₃ , ..., e _{4n}] |
| 5 | [e ₅₁ , e ₅₂ , e ₅₃ , ..., e _{5n}] |

Tabel 9 memuat nilai vektor untuk masing-masing kata. Masing-masing indeks kata pada teks *sequence* di Tabel 8 kemudian dilakukan pencarian atau pencocokan dengan vektor *word embedding*. Vektor *word embedding* yang sesuai dengan teks *sequence* dirata-rata. Nilai rata-rata pada setiap vektor tersebut kemudian digunakan sebagai input *neuron* pada *neural network* untuk dilakukan klasifikasi.

Klasifikasi pada penelitian ini menggunakan algoritma DNN dengan *pre-trained word embedding* FastText yang memiliki dimensi *embedding* sejumlah 300 dimensi.

2.8 Eksperimen dan Pengujian Model

Eksperimen model dilakukan dengan kombinasi jumlah *layer* dan jumlah *neuron* yang berbeda-beda. Pengujian model dilakukan dengan menghitung akurasi pada *training set* dan *testing set*. Akurasi merupakan perhitungan total klasifikasi sentimen yang benar terhadap seluruh data yang ditunjukkan pada Persamaan 6. Tabel 10 merupakan contoh data beserta klasifikasinya.

$$\text{Akurasi} = \frac{\text{Jumlah klasifikasi benar}}{\text{Jumlah dokumen uji}} \times 100 \% \quad (6)$$

Tabel 10. Contoh Data dan Kelas Prediksinya

| Data ke- | Actual Class | Predicted Class |
|----------|--------------|-----------------|
| 1 | Positif | Netral |
| 2 | Positif | Positif |
| 3 | Positif | Positif |
| 4 | Negatif | Negatif |
| 5 | Negatif | Negatif |
| 6 | Negatif | Negatif |
| 7 | Netral | Positif |
| 8 | Netral | Negatif |
| 9 | Netral | Negatif |
| 10 | Netral | Netral |

Pada Tabel 10 diperoleh 6 data dengan klasifikasi yang benar dan 4 data klasifikasi yang salah. Hasil tersebut dihitung akurasi dengan Persamaan 6:

$$\text{Akurasi} = \frac{6}{10} \times 100 \% = 60 \%$$

Dari perhitungan tersebut, nilai akurasi untuk contoh data pada Tabel 10 sebesar 60%.

3. HASIL

3.1 Pengumpulan Data

Pengambilan data dilakukan dengan menggunakan *Twitter API*. Untuk dapat menggunakan API tersebut maka dibutuhkan akun *Twitter developer* dengan melakukan registrasi ke URL <https://developer.twitter.com/en/apply-for-access> dan mengisi formulir pendaftaran dan ketentuan yang ditetapkan oleh pihak *Twitter*. Akun tersebut digunakan untuk mendapatkan API key serta *access token* yang merupakan kunci untuk mengakses *Twitter API*. Kedua kunci pengaksesan tersebut dimasukkan pada kode program untuk melakukan pengambilan data dengan menggunakan bahasa pemrograman Python. Selanjutnya digunakan pustaka *Tweepy* untuk mempermudah pengaksesan *Twitter API* dalam pengambilan data. Gambar 8 merupakan potongan kode program untuk melakukan *crawling* data *Twitter* menggunakan *Tweepy*.

```
import tweepy

auth = tweepy.OAuthHandler('API_KEY')
auth.set_access_token('ACCESS_TOKEN')
api = tweepy.API(auth)

for tweet in tweepy.Cursor(api.search, q=keyword,
result_type='recent', tweet_mode='extended', since_id=False,
lang='id').items():
```

Gambar 8. Kode Program Tweepy

Pada kode program di atas, *tweet* yang diambil merupakan *tweet* berbahasa Indonesia yang diatur dengan parameter *lang*. Pengumpulan data dilakukan antara bulan Juni 2020 sampai November 2020 dan didapatkan 5504 *record* data. Data yang diambil kemudian disimpan kedalam *file* berekstensi CSV dengan format penyimpanan seperti pada Tabel 11. Kolom data yang digunakan untuk proses selanjutnya adalah kolom *full_text* yang berisi konten dari suatu *tweet*.

Tabel 11. Format Penyimpanan CSV

| Kolom | Deskripsi |
|-------------------------|--|
| <i>userid</i> | ID unik dari pengguna <i>Twitter</i> yang menulis suatu <i>tweet</i> . |
| <i>user_screen_name</i> | Username dari akun <i>Twitter</i> pengguna yang menulis suatu <i>tweet</i> . |
| <i>tweet_id</i> | ID unik dari suatu <i>tweet</i> . |
| <i>created_at</i> | Tanggal dan jam dibuatnya <i>tweet</i> . |
| <i>full_text</i> | Isi lengkap dari suatu <i>tweet</i> . |
| <i>keyword</i> | Kata kunci penelusuran <i>tweet</i> |
| <i>scraping_date</i> | Tanggal dan waktu pengambilan <i>tweet</i> . |

3.2 Text Preprocessing

Sebelum proses klasifikasi diimplementasikan, perlu dilakukan *preprocessing* pada data set. *Preprocessing* ini bertujuan untuk mengurangi *noise* pada data set serta mengubah data set menjadi bentuk yang lebih terstruktur. Tahapan yang dilakukan pada *preprocessing* ini adalah *cleansing*, *case folding*, formalisasi, *stemming*, dan tokenisasi.

3.3 Labeling

Proses *labeling* dilakukan menggunakan aplikasi Sentiment Strength Detection dari hasil penelitian yang dilakukan oleh [13]. Pada aplikasi tersebut dihasilkan tiga kelas sentimen yaitu positif, negatif, dan netral. Data yang telah dilakukan *text preprocessing* berjumlah 3902 *record* diberikan label dengan aplikasi tersebut dan didapatkan hasil sebagai berikut.

Tabel 12. Jumlah Data Tiap Kelas pada Data set

| Kelas | Jumlah |
|---------|--------|
| Positif | 1219 |
| Netral | 1750 |
| Negatif | 933 |

3.4 Pembentukan Data Training dan Data Testing

Setelah *preprocessing* dan *labeling* selesai dilakukan maka diperoleh data set siap untuk diproses. Data set terlebih dahulu dibagi menjadi dua yaitu data *training* dan data *testing*. Data *training* merupakan data yang digunakan untuk melatih dan membuat model sedangkan data *testing* digunakan untuk menguji performa model.

Eksperimen pada penelitian ini dilakukan dengan dua skenario pembagian data. Pertama, dengan proporsi data 90:10 dan skenario kedua dengan proporsi data 80:20.

3.4.1 Pembagian Data pada Skenario Pertama

Pada skenario pertama, data *training* dan data *testing* dibagi dengan proporsi 90:10. Sebanyak 90% dari total data set atau 3512 digunakan sebagai data *training* dan 10% sisanya atau 390 data digunakan sebagai data *testing*. Tabel 13 menunjukkan

jumlah data pada masing-masing kelas yang ada pada data set *training*.

Dari total 3512 data yang ada pada data *training*, jumlah data yang memiliki kelas positif adalah 1069 data, jumlah data dengan kelas netral adalah sejumlah 1610 data dan kelas negatif memiliki 833 data.

Tabel 13. Jumlah Data pada Data *Training* Skenario Pertama

| Kelas | Jumlah |
|---------|--------|
| Positif | 1069 |
| Netral | 1610 |
| Negatif | 833 |

Sebanyak 10% dari data set dijadikan sebagai data *testing*. Tabel 14 menunjukkan jumlah data pada masing-masing kelas yang ada pada data set *testing*.

Tabel 14. Jumlah Data pada Data *Testing* Skenario Pertama

| Kelas | Jumlah |
|---------|--------|
| Positif | 150 |
| Netral | 140 |
| Negatif | 100 |

Dari total 390 data yang ada pada data *testing*, jumlah data yang memiliki kelas positif adalah 150 data, jumlah data dengan kelas netral adalah sejumlah 140 data, dan kelas negatif memiliki 100 data.

3.4.2 Pembagian Data pada Skenario Kedua

Pada skenario kedua, data *training* dan data *testing* dibagi dengan proporsi 80:20. Sebanyak 80% dari total data set atau 3122 digunakan sebagai data *training* dan 10% sisanya atau 780 data digunakan sebagai data *testing*. Tabel 15 merupakan jumlah data pada masing-masing kelas yang ada pada data data *training*.

Tabel 15. Jumlah Data pada Data *Training* Skenario Kedua

| Kelas | Jumlah |
|---------|--------|
| Positif | 929 |
| Netral | 1460 |
| Negatif | 733 |

Dari total 3122 data yang ada pada data *training*, jumlah data yang memiliki kelas positif adalah 929 data, jumlah data dengan kelas netral adalah sejumlah 1460 data dan kelas negatif memiliki 733 data. Sebanyak 10% dari data set tersebut dijadikan sebagai data *testing*.

Tabel 16. Jumlah Data pada Data *Testing* Skenario Kedua

| Kelas | Jumlah |
|---------|--------|
| Positif | 290 |
| Netral | 290 |
| Negatif | 200 |

Dari total 780 data yang ada pada data *testing*, jumlah data yang memiliki kelas positif adalah 290 data, jumlah data dengan kelas netral adalah sejumlah 290 data dan kelas negatif memiliki 200 data.

3.5 Klasifikasi dengan Deep Neural Network

Setelah data *training* dan data *testing* terbentuk maka selanjutnya dilakukan proses klasifikasi. Untuk dapat menggunakan <https://doi.org/10.25077/TEKNOSI.v7i2.2021.108-118>

algoritma *deep neural network* dalam melakukan analisis sentimen maka tahap pertama yang dibutuhkan adalah membuat *vocabulary* dan membuat *sequence* dari setiap kalimat yang ada pada data set. Gambar 9 merupakan implementasi kode dari pembentukan *vocabulary* dan *sequence* yang digunakan untuk proses klasifikasi DNN.

```

from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer(num_words=MAX_NB_WORDS, oov_token='<OOV>')
tokenizer.fit_on_texts(text)

word_index = tokenizer.word_index
vocab = len(word_index)+1

# TRAINING DATA
sequences_training = tokenizer.texts_to_sequences(textTr)
padded_training = pad_sequences(sequences_training, maxlen=max_kata,
padding='post')

# TEST DATA
sequences_test = tokenizer.texts_to_sequences(textTe)
padded_test = pad_sequences(sequences_test, maxlen=max_kata, padding='post')

```

Gambar 9. Implementasi *Vocabulary* dan *Text Sequence*

Pembentukan *vocabulary* dan *text sequence* dilakukan menggunakan *library* Tensorflow. *Vocabulary* merupakan daftar seluruh kata yang ada pada data set yang dibuat dengan memanfaatkan hasil *tokenizer*. *Sequence* kemudian dibuat sebanyak dua kali yaitu *sequence* untuk data set *training* dan *sequence* untuk data set *testing*. Pada *sequences* tersebut, setiap kalimat diberikan *padding post* yang bertujuan untuk menyamakan jumlah token kata yang diklasifikasikan dengan algoritma DNN.

Setelah *sequences* terbentuk maka langkah selanjutnya adalah memuat model FastText *pre-trained word embedding*. Model *word embedding* FastText yang berbahasa Indonesia memiliki dimensi vektor sebanyak 300 dimensi untuk setiap token katanya [15]. Berikut merupakan gambar implementasi kode untuk memuat *word embedding* FastText agar dapat digunakan dengan algoritma DNN.

```

# load fasttext
embeddings_index = {}
f = codecs.open('wiki.id.vec', encoding='utf-8')
for line in tqdm(f):
    values = line.rstrip().rsplit(' ')
    word = values[0]
    coefs = np.asarray(values[1:], dtype='float32')
    embeddings_index[word] = coefs
f.close()
print('found %s word vectors' % len(embeddings_index))

# build embedding matrix
words_not_found = []
nb_words = min(MAX_NB_WORDS, len(word_index))
embedding_matrix = np.zeros((nb_words, embed_dim))
for word, i in word_index.items():
    if i >= nb_words:
        continue
    embedding_vector = embeddings_index.get(word)
    if (embedding_vector is not None) and len(embedding_vector) > 0:
        embedding_matrix[i] = embedding_vector
    else:
        words_not_found.append(word)

```

Gambar 10. Implementasi Kode *Word Embedding*

Hasil keluaran dari kode di atas adalah matriks yang berisi nilai vektor *word embedding* untuk setiap token kata pada data set. Token kata pada data set yang tidak ditemukan pada vektor *word embedding* FastText diberi nilai nol sedangkan token kata pada

data set yang terindeks pada *word embedding* FastText diberi nilai vektor sesuai dengan nilai yang ada pada *word embedding* tersebut. Hasil matriks tersebut kemudian digunakan sebagai *layer embedding* pada algoritma DNN.

3.5.1 Tahap Pelatihan dan Pengujian Model

Tahap pelatihan model dilakukan menggunakan algoritma DNN dengan variasi jumlah *hidden layer*, susunan node, dan nilai *learning rate*. Untuk setiap model DNN, *layer* pertama merupakan *layer* yang berisi *embedding* dan *layer* terakhir merupakan *output layer* yang berisi tiga *node* dan menggunakan fungsi aktivasi *softmax*, di antara *layer* pertama dan terakhir disisipkan *hidden layer* dengan jumlah yang bervariasi.

Tabel 17. Hasil Eksperimen Skenario Pertama

| Nama Model | Hid- den La- yer | Susunan Node Tiap Hidden Layer | Learn- -ing Rate | Train- ing Accu- racy | Testing Accu- racy |
|------------|---------------------------|--|------------------------|--------------------------------|--------------------------|
| DNN_01 | 1 | Dense(64) | 0.01 | 95.90% | 85.13% |
| DNN_02 | 1 | Dense(64) | 0.005 | 94.39% | 86.67% |
| DNN_03 | 1 | Dense(128) | 0.01 | 95.73% | 85.64% |
| DNN_04 | 1 | Dense(128) | 0.005 | 95.13% | 86.67% |
| DNN_05 | 1 | Dense(256) | 0.01 | 95.33% | 84.87% |
| DNN_06 | 1 | Dense(256) | 0.005 | 93.05% | 85.38% |
| DNN_07 | 2 | Dense(64) Dense(128) | 0.01 | 95.73% | 85.13% |
| DNN_08 | 2 | Dense(64) Dense(128) | 0.005 | 95.79% | 86.41% |
| DNN_09 | 2 | Dense(256) Dense(64) | 0.01 | 96.10% | 86.67% |
| DNN_10 | 2 | Dense(256) Dense(64) | 0.005 | 94.99% | 87.69% |
| DNN_11 | 3 | Dense(128) Dense(256) Dense(128) | 0.01 | 96.05% | 87.95% |
| DNN_12 | 3 | Dense(128) Dense(256) Dense(128) | 0.005 | 95.30% | 88.72% |
| DNN_13 | 3 | Dense(256) Dense(128) Dense(128) | 0.005 | 96.75% | 87.18% |
| DNN_14 | 3 | Dense(128) Dense(128) Dense(128) | 0.005 | 95.96% | 87.18% |
| DNN_15 | 4 | Dense(64) Dense(128) Dense(256) Dense(128) | 0.005 | 95.44% | 84.62% |
| DNN_16 | 4 | Dense(64) Dense(128) Dense(256) Dense(128) | 0.0005 | 95.30% | 85.13% |
| DNN_17 | 4 | Dense(256) Dense(128) Dense(128) Dense(256) | 0.005 | 94.22% | 86.15% |
| DNN_18 | 5 | Dense(128) Dense(128) Dense(128) Dense(256) Dense(256) | 0.005 | 95.96% | 86.41% |
| DNN_19 | 5 | Dense(128) Dense(128) Dense(128) Dense(256) Dense(256) | 0.0005 | 95.10% | 88.21% |
| DNN_20 | 5 | Dense(256) Dense(128) Dense(128) Dense(256) Dense(256) | 0.005 | 95.67% | 87.44% |

Eksperimen dilakukan dengan dua skenario yaitu skenario pertama dengan proporsi data set 90:10. Sebanyak 3512 data *training* dan 390 data *testing* digunakan pada proses pelatihan dan pengujian. Tabel 17 menunjukkan hasil eksperimen pelatihan dan pengujian model DNN untuk skenario pertama. Adapun hasil eksperimen dari skenario kedua dengan proporsi data *training* sebanyak 3112 dan 780 data *testing*, disajikan dalam Tabel 18.

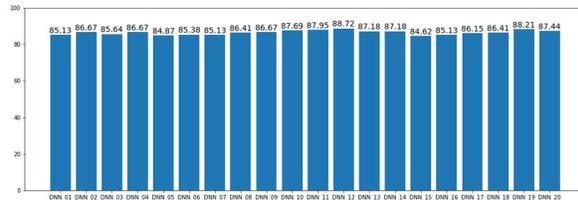
Tabel 18. Hasil Eksperimen Skenario Kedua

| Nama Model | Hid- den La- yer | Susunan Node Tiap Hidden Layer | Learn- -ing Rate | Train- ing Accu- racy | Testing Accu- racy |
|------------|---------------------------|--|------------------------|--------------------------------|--------------------------|
| DNN_21 | 1 | Dense(64) | 0.01 | 91.38% | 82.95% |
| DNN_22 | 1 | Dense(64) | 0.005 | 91.38% | 83.46% |
| DNN_23 | 1 | Dense(128) | 0.01 | 90.20% | 82.18% |
| DNN_24 | 1 | Dense(128) | 0.005 | 89.78% | 83.21% |
| DNN_25 | 2 | Dense(64) Dense(128) | 0.01 | 91.80% | 76.67% |
| DNN_26 | 2 | Dense(64) Dense(128) | 0.005 | 89.97% | 81.97% |
| DNN_27 | 2 | Dense(128) Dense(128) | 0.01 | 90.26% | 81.41% |
| DNN_28 | 2 | Dense(128) Dense(256) | 0.01 | 91.13% | 80.00% |
| DNN_29 | 3 | Dense(128) Dense(256) Dense(128) | 0.01 | 93.24% | 82.18% |
| DNN_30 | 3 | Dense(128) Dense(256) Dense(128) | 0.005 | 91.93% | 82.56% |
| DNN_31 | 3 | Dense(128) Dense(256) Dense(256) | 0.01 | 91.06% | 83.59% |
| DNN_32 | 3 | Dense(64) Dense(128) Dense(256) | 0.01 | 91.32% | 82.95% |
| DNN_33 | 4 | Dense(64) Dense(128) Dense(256) Dense(128) | 0.01 | 92.70% | 83.08% |
| DNN_34 | 4 | Dense(64) Dense(128) Dense(256) Dense(128) | 0.005 | 90.71% | 83.33% |
| DNN_35 | 4 | Dense(128) Dense(128) Dense(128) Dense(256) | 0.01 | 90.58% | 80.90% |
| DNN_36 | 4 | Dense(256) Dense(128) Dense(128) Dense(128) | 0.01 | 90.97% | 79.36% |
| DNN_37 | 5 | Dense(128) Dense(128) Dense(128) Dense(256) Dense(256) | 0.01 | 91.61% | 76.67% |
| DNN_38 | 5 | Dense(128) Dense(128) Dense(128) Dense(256) Dense(256) | 0.005 | 90.65% | 82.56% |
| DNN_39 | 5 | Dense(256) Dense(256) Dense(128) Dense(128) Dense(128) | 0.01 | 89.30% | 78.97% |
| DNN_40 | 5 | Dense(256) Dense(256) Dense(128) Dense(128) Dense(64) | 0.01 | 90.49% | 81.15% |

4. PEMBAHASAN

4.1 Analisis Eksperimen Skenario Pertama

Eksperimen pertama dengan 80:20 perbandingan data *training* dan *testing* dilakukan dengan menggunakan 20 model yang bervariasi parameternya. Hasilnya, pada Gambar 11 dapat ditunjukkan bahwa model DNN_12 memperoleh akurasi tertinggi dengan persentase 88.72%. Hasil tersebut diperoleh dari model yang tersusun oleh 3 *hidden layer* dengan jumlah masing-masing *node*-nya berturut-turut adalah 128, 256, dan 128, serta menggunakan *learning rate* bernilai 0.005.



Gambar 11. Akurasi Setiap Model Skenario Pertama

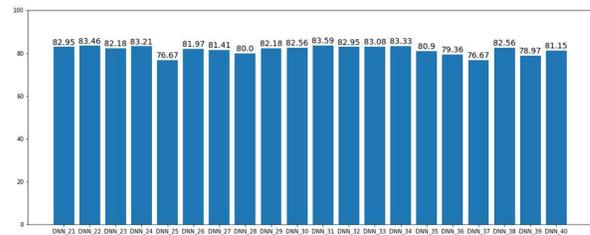
Pada Tabel 17 dapat diperhatikan bahwa semakin banyak jumlah *hidden layer* maka nilai akurasi yang didapatkan tidak selalu lebih baik. Contohnya pada model DNN_03 dan DNN_04 yang keduanya memiliki 1 *hidden layer* dan menghasilkan nilai akurasi yang lebih baik dari model DNN_15 dan DNN_16 yang memiliki 4 *hidden layer*.

Selain itu, dari 20 model yang ada pada Tabel 17 juga ditunjukkan bahwa nilai *learning rate* berpengaruh terhadap akurasi model. Pada model dengan jumlah *hidden layer* dan susunan *node* yang sama maka nilai *learning rate* yang lebih kecil memiliki nilai akurasi yang lebih besar dari model lainnya. Contohnya pada model DNN_18 dan DNN_19. Kedua model tersebut sama-sama memiliki 5 *hidden layer* dengan susunan *node* 128, 128, 128, 256, dan 256 *node*. Model DNN_18 dengan nilai *learning rate* sebesar 0.005 menghasilkan akurasi sebesar 86.41% sedangkan model DNN_19 yang memiliki *learning rate* sebesar 0.0005 menghasilkan akurasi sebesar 88.21%.

Kemudian hasil akurasi pada model yang memiliki parameter jumlah *hidden layer*, dan *learning rate* yang sama namun dengan jumlah *node* yang berbeda maka semakin banyak jumlah *node* tidak menjamin semakin besarnya akurasi yang dihasilkan. Contohnya seperti pada model DNN_01, DNN_03, DNN_5 yang ketiganya memiliki 1 *hidden layer* dan *learning rate* 0.01 dengan jumlah *node* untuk setiap model berturut-turut sebanyak 64, 128, dan 256 *node*. Akurasi yang dihasilkan dari model tersebut bervariasi dan tidak bergantung pada besarnya jumlah *node* pada *layer* tersebut.

4.2 Analisis Eksperimen Skenario Kedua

Sebanyak 3112 data *training* dan 780 data *testing* digunakan pada proses pelatihan dan pengujian ini.



Gambar 12. Akurasi Setiap Model Skenario Kedua

Dari eksperimen pada Tabel 18 maka dapat ditunjukkan bahwa semakin banyak jumlah *layer* maka tidak menjamin model mendapatkan akurasi yang lebih besar. Contohnya pada model DNN_21 dengan 1 *hidden layer* memiliki akurasi yang lebih tinggi yaitu 82.95% dari model DNN_26 dengan 2 *hidden layer* yang memiliki akurasi 81.97%.

Di sisi lain, untuk model dengan susunan *node* yang sama maka *learning rate* yang lebih kecil memiliki nilai akurasi yang lebih baik. Contohnya pada model DNN_29 dan DNN_30 yang masing-masing memiliki susunan *node* yang sama. Model DNN_29 yang mempunyai *learning rate* 0.01 memperoleh akurasi yang lebih kecil yaitu 82.18% dari model DNN_30 dengan *learning rate* 0.005 yang memiliki akurasi 82.56%.

Dari grafik hasil 20 model pelatihan skenario kedua pada Gambar 12, maka dapat disimpulkan bahwa model terbaik ada pada model DNN_31 yang memiliki 3 *hidden layer* dengan susunan *node* 128, 256, 256 *node*. *Learning rate* yang digunakan pada model DNN_31 adalah 0.01 dan menghasilkan hasil akurasi sebesar 83.59%. Namun, performa model terbaik pada skenario kedua mendapatkan akurasi yang lebih kecil dari performa model terbaik pada skenario pertama yang ditunjukkan pada Tabel 17 yaitu model DNN_12 dengan nilai akurasi sebesar 88.72%.

Angka tersebut menunjukkan akurasi yang lebih baik jika dibandingkan dengan performa metode SVM (RBF) dan K-NN pada [7], dengan akurasi tertinggi 87% dan 82% berturut-turut. Hasil ini dapat dijadikan sebagai referensi untuk penelitian selanjutnya pada bidang analisis sentimen, atau penelitian sejenis.

5. KESIMPULAN

Analisis sentimen *Twitter* untuk menilai opini terhadap perusahaan publik menggunakan algoritma DNN berhasil diterapkan. Tahap awal penelitian dilakukan dengan pengumpulan data *tweet* yang diambil dengan *Twitter API*, pada rentang Juni 2020 sampai November 2020 dan didapatkan 5504 *record* data. Data tersebut kemudian dilakukan *preprocessing* berupa *cleansing*, *case folding*, formalisasi, *stemming*, dan tokenisasi. Selanjutnya pelabelan diselesaikan menggunakan Sentiment Strength Detection, sehingga siap untuk dimodelkan dan diuji.

Algoritma DNN untuk klasifikasi *tweet* berbahasa Indonesia ini dilakukan secara otomatis berdasarkan tiga kelas yaitu positif, negatif, dan netral. Klasifikasi diterapkan menggunakan *pre-trained word embedding* FastText yang memiliki dimensi *embedding* sejumlah 300 dimensi.

Eksperimen dilakukan dengan dua skenario proporsi data *training* dan *testing*, pertama 90:10 dan kedua adalah 80:20. Eksperimen skenario pertama dengan total 3902 menghasilkan model DNN dengan performa terbaik dari seluruh eksperimen. Model tersebut tersusun dari 3 *hidden layer* dengan susunan *node* tiap *layer* pada model tersebut yaitu 128, 256, 128 *node* dan menggunakan *learning rate* sebesar 0.005. Nilai akurasi yang diperoleh dari model DNN tersebut adalah 88.72%.

DAFTAR PUSTAKA

- [1] C. Eforis, "Pengaruh Kepemilikan Negara Dan Kepemilikan Publik Terhadap Kinerja Keuangan Bumn (Studi Pada Perusahaan Bumn Yang Go Public Pada Tahun 2012 – 2015)," *Ultim. J. Ilmu Akunt.*, vol. 9, no. 1, pp. 18–31, 2017.
- [2] B. Nugraha, "Metode Klasifikasi Analisis Sentimen pada Media Sosial," *Syntax J. Inform.*, vol. 9, no. 2, pp. 109–118, 2020.
- [3] Z. Jianqiang, G. Xiaolin, and Z. Xuejun, "Deep Convolution Neural Networks for Twitter Sentiment Analysis," *IEEE Access*, vol. 6, pp. 23253–23260, 2018.
- [4] A. Rossi, T. Lestari, R. Setya Perdana, and M. A. Fauzi, "Analisis Sentimen Tentang Opini Pilkada DKI 2017 Pada Dokumen Twitter Berbahasa Indonesia Menggunakan Naïve Bayes dan Pembobotan Emoji," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 1, no. 12, pp. 1718–1724, 2017.
- [5] A. M. Pravina, I. Cholissodin, and P. P. Adikara, "Analisis Sentimen Tentang Opini Maskapai Penerbangan pada Dokumen Twitter Menggunakan Algoritme Support Vector Machine (SVM)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 3, no. 3, pp. 2789–2797, 2019.
- [6] U. Rofiqoh, R. S. Perdana, and M. A. Fauzi, "Analisis Sentimen Tingkat Kepuasan Pengguna Penyedia Layanan Telekomunikasi Seluler Indonesia Pada Twitter Dengan Metode Support Vector Machine dan Lexion Based Feature," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 1, no. 12, pp. 1725–1732, 2017.
- [7] R. Saputri, W. Winahju, and K. Fithriasari, "Klasifikasi Sentimen Wisatawan Candi Borobudur pada Situs TripAdvisor Menggunakan Support Vector Machine dan K-Nearest Neighbor," *J. Sains dan Seni ITS*, vol. 8, no. 2, pp. 349–356, 2019.
- [8] R. Cahyani and P. P. Adikara, "Analisis Sentimen terhadap Ulasan Hotel menggunakan Boosting Weighted Extreme Learning Machine," vol. 3, no. 8, pp. 7767–7773, 2019.
- [9] W. Parasati, F. A. Bachtiar, and N. Y. Setiawan, "Analisis Sentimen Berbasis Aspek pada Ulasan Pelanggan Restoran Bakso President Malang dengan Metode Naïve Bayes Classifier," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 4, no. 4, pp. 1090–1099, 2020.
- [10] M. A. Azim and M. H. Bhuiyan, "Text to emotion extraction using supervised machine learning techniques," *Telkonnika (Telecommunication Comput. Electron. Control.*, vol. 16, no. 3, pp. 1394–1401, 2018.
- [11] K. T. Chitty-Venkata and A. K. Somani, "Array aware training/pruning: Methods for efficient forward propagation on array-based neural network accelerators," *Proc. Int. Conf. Appl. Syst. Archit. Process.*, vol. 2020-July, pp. 37–44, 2020.
- [12] W. C. F. Mariel, S. Mariyah, and S. Pramana, "Sentiment analysis: A comparison of deep learning neural network algorithm with SVM and naïve Bayes for

Indonesian text," *J. Phys. Conf. Ser.*, vol. 971, no. 1, 2018.

- [13] D. H. Wahid and A. SN, "Peringkasan Sentimen Ekstraktif di Twitter Menggunakan Hybrid TF-IDF dan Cosine Similarity," *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 10, no. 2, p. 207, 2016.
- [14] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, pp. 1–25, 2018.
- [15] Facebook, "Wiki word vectors - fastText," 2017. .

NOMENKLATUR

| | |
|----------|-------------------------------|
| y | output |
| α | fungsi aktivasi |
| w | bobot |
| x | input |
| b | bias |
| e | bilangan alami |
| max | nilai maksimum |
| j | neuron ke-j pada <i>layer</i> |
| K | jumlah kelas |

BIODATA PENULIS

Erwin Yudi Hidayat

Menamatkan pendidikan sarjana di Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, sebagai Sarjana Komputer (S.Kom). Studi jenjang S2 diselesaikan di Universiti Teknikal Malaysia Melaka (UTeM), dengan konsentrasi pada bidang kecerdasan buatan. Penulis tertarik dalam penelitian *machine learning*, *deep learning*, dan pengolahan citra digital. Saat ini menjadi staf pengajar pada almamater di tempat kuliah S1 dulu ditempuh.

Raindy Wicaksana Hardiansyah

Penulis adalah alumni Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Semarang. Ketertarikan akan *text mining* dan kecerdasan buatan, mengantarkan penulis terhadap penelitian analisis sentimen. Gelar S.Kom yang diperoleh diselesaikan dalam waktu 3,5 tahun dan lulus dengan *cumlaude*.

Affandy

Penulis adalah staf pengajar aktif di Fakultas Ilmu Komputer, Universitas Dian Nuswantoro, Semarang, tempat program sarjana dan magister diselesaikan. Gelar *Doctor of Philosophy* diperoleh dari Universiti Teknikal Malaysia Melaka (UTeM), dengan minat penelitian pada *software visualization*, *software engineering*, dan *Computer Science Education*