BAB II

LANDASAN TEORI

2.1 Analisis Sentimen

Analisis sentimen merupakan proses yang mengotomasikan penambangan dan klasifikasi pendapat, pandangan, emosi, dan sentimen dataset teks yang tidak terstruktur dengan bahasa mesin dan pemrograman komputer (Fiarni dkk., 2016). Tugas dasar dalam analisis sentimen adalah mengelompokkan atau klasifikasi berdasarkan polaritas.

Berdasarkan klasifikasi, analisi sentimen dibagi menjadi dua kelompok utama, yaitu dokumen klasifikasi ke pendapat atau fakta atau dikenal sebagai klasifikasi subjektivitas (*subjectivity classification*) dan dokumen klasifikasi ke dalam positif atau negatif, atau dikenal sebaga analisis sentimen. Hal ini adalah proses yang penting untuk menentukan dokumen yang memiliki opini dan dokumen yang menyimpulkan opini bernilai positif, negatif maupun netral (Nurhuda, Sihwi and Doewes, 2013).

2.2 Text Classification

Menurut Nedjah (2009) *text classification* didefinisikan sebagai pengkategorian teks secara otomatis ke dalam satu atau lebih kelas yang telah ditentukan berdasarkan isinya. Hal ini dapat dilakukan dengan memberikan label kepada dokumen berdasarkan kumpulan dokumen sebelumnya yang sudah diberi label. Langkah pertama adalah merepresentasikan dokumen sebagai vektor dalam sebuah *high-dimension vector*, dimana masing-masing dimensi menyesuaikan

kepada nilai sebuah fitur, atau secara spesifik, sebuah *term*. Program kemudian menggunakan vektor tersebut untuk mengklasifikasikan dokumen ke kategori yang tepat. Misalkan pada koran jika ada isi berita mengenai kenaikkan tingkat suku bunga, atau penurunan bea masuk maka akan dimasukkan ke dalam artikel ekonomi.

Tujuan dari kategorisasi teks adalah menguji pengklasfikasian teks yang belum diketahui kategorinya, jadi jika ada teks yang baru dapat lebih mudah diklasfikasikan pada suatu kategori berdasarkan teks-teks yang telah ada sebelumnya (Gaikwad dkk., 2014). Menurut Aggarwal (2012) berbagai macam teknik telah dirancang untuk melakukan klasifikasi text salah satunya menggunakan pendekatan *machine learning*. Metode atau algoritma yang biasa digunakan untuk melakukan klasifikasi text antara lain *Decision Trees, Pattern (Rule)-based Classifiers, SVM Classifiers, Neural Network Classifiers,* dan *Bayesian (Generative) Classifiers.*

2.3 Word Embedding

Word Embedding adalah representasi vektor bernilai nyata dari kata-kata dengan menanamkan makna semantik dan sintaksis yang diperoleh dari *corpus* besar yang tidak berlabel. Merupakan alat yang ampuh dan banyak digunakan dalam *Natural Language Processing* (NLP), termasuk analisis semantik, pencarian informasi, *parsing* depedensi, penjawab pertanyaan, dan terjemahan bahasa mesin (Yu dkk., 2017).

Sebagai contoh untuk mengubah kalimat menjadi vektor, misalnya ada dua kalimat seperti ini, “have a good day” dan “have a great day”. Mereka hampir tidak

memiliki makna yang berbeda. Lalu akan dibuat sebuah kamus *vocabulary* yang berisikan kata kata unik dari kedua kalimat tersebut (sebut saja dengan V), menjadi seperti berikut, V = {have, a, good, great, day}. Lalu dengan metode *one-hot encoded vector* dengan ukuran yang sama dengan V (=5) akan menghasilkan vektor 0 untuk elemen di indeks yang mewakili kata yang sesuai dalam kosakata. Elemen kata itu akan menjadi satu. Hasil encoding menjadi seperti ini, have = [1,0,0,0,0]`; a = [0,1,0,0,0]`; good = [0,0,1,0,0]`; great = [0,0,0,1,0]`; day = [0,0,0,0,1]` (` merepresentasikan *transpose*).

Metode word embedding menggunakan teknik pembelajaran *neural network model* untuk merepresentasikan vektor dari kosakata yang konstan yang berasal dari kumpulan teks. Menurut Kaibi dkk. (2019) secara umum ada 3 model yang sering digunakan untuk melakukan word embedding, yaitu Word2Vec, GloVe (*Global Vector*), dan FastText.

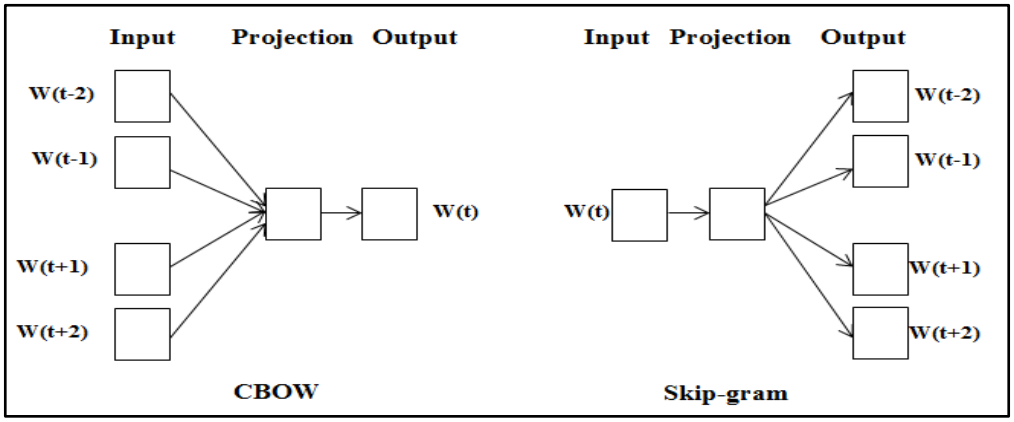
2.4 FastText

FastText adalah *library* yang dikeluarkan oleh Facebook yang merupakan pengembangan dari Word2Vec. FastText dan Word2Vec sama-sama dapat digunakan untuk *word embedding* dan mendeteksi *word similarity* secara semantik maupun sintaksis. Menurut Kamus Besar Bahasa Indonesia (KBBI) pengertian semantik adalah bagian struktur bahasa yang berhubungan dengan makna ungkapan atau struktur makna suatu wicara. Dalam hal ini *word similarity* secara semantik berarti ada kemiripan atau kedekatan dari segi makna kata. Sedangkan sintaksis pada KBBI artinya cabang linguistik tentang susunan kalimat dan bagiannya,

artinya apabila *word similarity* secara sintaksis maka terdapat kemiripan atau kedekatan dari segi susunan atau struktur pada kata (Rehurek, 2019).

Melalui FastText, input kata akan direpresentasikan dalam bentuk vektor dengan urutan penempatan sedemikian rupa sehingga kata-kata yang bermakna sama akan muncul bersamaan, dan yang berbeda akan berjauhan. FastText dapat digunakan untuk memperoleh vektor untuk kata-kata *out-of-vocabulary* (OOV) (Rehurek, 2019).

Terdapat dua jenis arsitektur di dalam FastText yang mirip dengan Word2Vec yaitu *Contious Bag of Words* (CBOW) dan Skip-gram. Perbedaan CBOW dan Skip-gram yaitu, cara CBOW bekerja adalah cenderung memprediksi probabilitas kata (*current word*) sebagai target yang diberikan konteks sebagai input. Sedangkan Skip-gram merupakan kebalikannya dari CBOW. Konteks dapat berupa satu kata atau sekelompok kata. Visualisasi dari arsitektur CBOW dan Skip-gram dapat dilihat pada Gambar 2.1.



Gambar 2.1 Arsitektur CBOW dan Skip-gram
(Sumber: Mikolov, 2013)

Rehurek juga berpendapat bahwa diperlukan *dataset* tersendiri untuk memperoleh *dictionary* yang diinginkan, biasa disebut *embedding dataset*.

Embedding dataset diambil dari *train set* setelah *dataset* utama dilakukan split. *Embedding dataset* yang telah di-*training* menggunakan FastText akan menghasilkan *vocabulary* yang berisikan kosa kata yang dapat digunakan untuk mendeteksi *word similarity*. Hasil *training* ini disebut *pre-trained model* (Rehurek, 2019).

2.5 TF-IDF

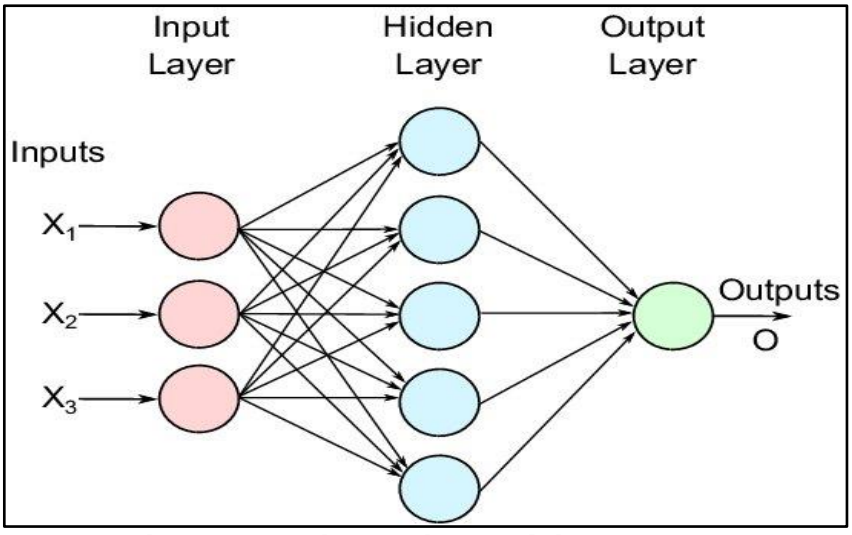
Term Frequency-Inverse Document Frequency atau TF-IDF merupakan algoritma yang berguna untuk mengetahui bobot setiap kata atau seberapa sering kata tersebut. Metode ini menggabungkan pehitungan bobot, yaitu frekuensi kemunculan sebuah kata di dalam sebuah dokumen (*Term Frequency*) tertentu dan *inverse* frekuensi dokumen yang mengandung kata tersebut (*Inverse Document Frequency*) (Nurjannah dkk., 2013). Frekuensi kemunculan kata (*Term Frequency*) di dalam dokumen menunjukkan seberapa penting kata tersebut di dalam dokumen tersebut. Frekuensi dokumen yang mengandung kata tersebut *(Inverse Document Frequency*) menunjukkan seberapa umum kata tersebut, sehingga bobot hubungan antara sebuah kata dan sebuah dokumen akan tinggi apabila frekuensi kata tersebut tinggi di dalam dokumen dan frekuensi keseluruhan dokumen yang mengandung kata tersebut yang rendah pada kumpulan dokumen (Nurjannah dkk., 2013).

Di dalam ekstraksi fitur TF-IDF terdapat n-gram. N-gram adalah potongan sejumlah n kata dari sebuah kalima. Digunakan untuk mengambil potongan kata sejumlah n dari kalimat yang secara kontinuitas dibaca dari teks sumber hingga akhir dokumen. N-gram dibedakan berdasarkan jumlah potongan kata sebesar n. contoh kata “The cow jumps over” dapat diuraikan ke beberapa n-gram berikut:

- Uni-grams : The, cow, jumps, over
- Bi-grams: The cow, cow jumps, jumps over
- Tri-grams: The cow jumps, cow jumps over

2.6 Multilayer Perceptron

Multilayer Perceptron (MLP) merupakan ANN turunan dari *Perceptron*, berupa ANN umpan balik (*feedforward*) dengan satu atau lebih *layer* tersembunyi (*hidden layer*) yang memetakan set data input ke output yang sesuai (Jotheeswaran and Koteeswaran, 2015). Biasanya, jaringan terdiri atas satu *layer* masukan, setidaknya satu *layer neuron* komputasi di tengah (tersembunyi), dan sebuah *layer neuron* komputasi keluaran. Sinyal masukan ditambatkan dengan arah maju pada *layer-per-layer*. Gambar 2.2 merupakan contoh arsitektur MLP.



Gambar 2.2 Contoh Arsitektur Multilayer Perceptron
(sumber: www.researchgate.net)

Pada gambar tersebut ada satu *layer* tersembunyi dengan lima *neuron*, dan satu *layer* keluaran dengan satu *neuron*. Sebenarnya ada satu *layer* lagi dalam MLP, yaitu *layer* memasukan berupa vektor masukan. Namun, dalam *layer* ini tidak ada

komputasi, yang dilakukan hanya meneruskan sinyal/vektor masukan yang diterima ke *layer* di depannya. Setiap *layer* dalam MLP mempunyai fungsi khusus. Layer masukkan berfungsi menerima sinyal/vektor, layer keluaran menerima sinyal keluaran (atau dengan kata lain, stimulus pola) dari layer tersembunyi dan memunculkan sinyal/nilai/kelas keluaran dari keseluruhan jaringan.

Neuron dalam *layer* tersembunyi mendeteksi fitur-fitur tersembunyi. Bobot dari *neuron* dalam *layer* tersembunyi merepresentasikan fitur tersembunyi dalam vektor masukan. Fitur-fitur tersembunyi ini kemudian digunakan oleh *layer* keluaran dalam penentuan pola/kelas keluaran.

ANN yang sering digunakan biasanya terdiri atas tiga atau bahkan empat layer. Termasuk satu atau dua layer tersembunyi. Setiap layer bisa berisi 10 sampai 1.000 neuron, tetapi dalam kebanyakan aplikasi menggunakan satu atau dua layer tersembunyi karena setiap penambahan satu layer akan meningkatkan beban komputasi secara eksponensial (Prasetyo, 2012). Berikut adalah algoritma dari Multilayer Perceptron (Muliantara dan Widiartha, 2011).

1. Inisialisasi semua bobot dengan bilangan acak kecil.
2. Jika kondisi penghentian belum dipenuhi, lakukan langkah 2-8.
3. Untuk setiap pasang data pelatihan, lakukan langkah 3-8.
4. Tiap unit masukan menerima sinyal dan meneruskan ke unit tersembunyi di atasnya.
5. Hitung semua keluaran di unit tersembunyi z_j $(j = 1, 2, \dots, p)$.

$$z\_net_j = v_{j0} + \sum_{i=1}^{n} x_i \, v_{ji} \qquad \dots(2.1)$$

$$z_i = f(z\_net_j) = \frac{1}{1 + e^{-z\_net_j}} \quad \ldots(2.2)$$

6. Hitung semua keluaran jaringan di unit keluaran y_k $(k = 1, 2, \ldots, m)$.

$$y\_net_k = w_{k0} + \sum_{j=1}^{p} z_j w_{kj} \quad \ldots(2.3)$$

$$y_k = f(y\_net_k) = \frac{1}{1 + e^{-y\_net_k}} \quad \ldots(2.4)$$

7. Hitung faktor δ unit keluaran berdasarkan kesalahan di setiap unit keluaran y_k $(k = 1, 2, \ldots, m)$.

$$\delta_k = (t_k - y_k) f'(y_{net_k}) = (t_k - y_k) y_k (1 - y_k) \quad \ldots(2.5)$$

$$t_k = target$$

δk merupakan unit kesalahan yang akan dipakai dalam perubahan bobot layer di bawahnya. Hitung perubahan bobot w_{kj} dengan laju pemahaman α.

$$\triangle w_{kj} = \alpha \delta_k Z_j \quad \ldots(2.6)$$

$$k = 1,2, \ldots, m; \quad j = 0,1, \ldots, p$$

8. Hitung faktor δ unit tersembunyi berdasarkan kesalahan di setiap unit tersembunyi $z_j (j = 1)$.

$$\delta\_net_j = \sum_{k=1}^{m} \delta_k w_{kj} \quad \ldots(2.7)$$

Faktor δ unit tersembunyi.

$$\delta_j = \delta\_net_j f'\left(Z_{net_j}\right) = \delta\_net_j z_j (1 - z_j) \quad \ldots(2.8)$$

Hitung suku perubahan bobot v_{ij}

$$\triangle v_{ji} = \alpha\delta_j x_i \quad \ldots(2.9)$$

$$j = 1,2,\ldots,p; \quad i = 1,2,\ldots,n$$

9. Hitung semua perubahan bobot. Perubahan bobot garis yang menuju ke uit keluaran yaitu:

$$w_{kj}(baru) = w_{kj}(lama) + \triangle w_{kj} \quad \ldots(2.10)$$

$$(k = 1,2,\ldots,m; j = 0,1,\ldots,p)$$

Perubahan bobot garis yang menuju ke unit tersembunyi yaitu:

$$V_{ji}(baru) = v_{ji}(lama) + \triangle v_{ji} \quad \ldots(2.11)$$

$$(j = 1,2,\ldots,p; i = 0,2,\ldots,n)$$

2.7 Evaluasi Performa

Evaluasi performa algoritma Multilayer Perceptron dengan menggunakan metode word embedding yang dilakuakan dalam penelitian ini diukur dengan menghitung accuracy, Precision, Recall, dan F1. Perhitungan untuk evaluasi performa dapat dilihat pada persamaan berikut.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad \ldots(2.12)$$

$$Precision = \frac{TP}{(TP + FP)} \quad \ldots(2.13)$$

$$Recall = \frac{TP}{(TP + FN)} \quad \ldots(2.14)$$

$$F1 = \frac{(2 \times Precision \times Recall)}{(Precision + Recall)} \quad \ldots(2.15)$$

TP, TN, FP, dan FN adalah nilai dari true positive, true negative, false positive, dan false negative yang didapat dari *Confussion Matrix*. Confussion

Matrix dikenal juga sebagai tabel kontingensi yang biasanya digunakan dalam *supervised learning* untuk memungkinkan visualisasi kinerja algoritma yang digunakan (Mouthami, Devi and Bhaskaran, 2013) secara umum dapat dilihat pada Tabel 2.1.

Tabel 2.1 *Confusion Matrix*

Prediksi / **Aktual**	**Negatif**	**Positif**
Negatif	TN = True Negative	FP = False Positive
Positif	FN = False Negative	TP = True Positive

2.8 Media Sosial Twitter

Media sosial menawarkan sumber data yang berharga. Posting pengguna di media sosial dapat membertikan wawasan tentang pendapat, perilaku, Gerakan, status kesehatan, dan detail lainnya tentang kehidupan mereka. Mengumpulkan dan menganalusus sejumlah besar data ini dapat digunakan dalam berbagai aplikasim termasuk opini masyarakat. Twitter adalah salah satu platfor, media sosial yang paling cepat berkembang di dunia. Pengguna dapat memposting pesan (tweet) di profil mereka hingga 280 karakter per tweet. Selain itu pesan dalam Twitter bersifat publik dan tidak mengharuskan memberi ijin kepada orang lain untuk melihat apa yang kita tulis dan dapat dicari melalui fungsi pencarian Twitter (Alshammari dan Nielsen, 2018).