BAB 2

TINJAUAN PUSTAKA

2.1 Text Classification dan Text Processing

Text classification adalah kegiatan memberi label dengan kategori tertentu yang sudah ditetapkan pada suatu *text* dengan bahasa alami (Sebastiani, 2002). Pada saat ini, *text classification* diterapkan dalam berbagai konteks, mulai dari pengindeksan dokumen yang berdasarkan kosa kata yang terkontrol, penyaringan dokumen, pembuatan metadata otomatis, ataupun berbagai aplikasi lain yang membutuhkan organisasi dokumen (Sebastiani, 2002). Ada beberapa strategi yang umum dalam *text classification* secara otomatis, yaitu dengan *pre-processing, feature extraction/ selection*, *modeling* dengan menggunakan teknik pembelajaran mesin yang sesuai, serta *training* dan *testing* pada *classifier* (K. Dalal & A. Zaveri, 2011).

Tahap *pre-processing* adalah proses untuk mempersiapkan data mentah sebelum dilakukan proses lain. Pada umumnya, *pre-processing* data dilakukan dengan cara mengeliminasi data yang tidak sesuai atau mengubah data menjadi bentuk yang untuk menjadi yang lebih mudah diproses oleh sistem (Mujilahwati, 2016). Menurut Marfian (2015) terdapat beberapa tahapan dalam *text pre-processing* sebagai berikut :

1. Case Folding

Pada tahapan ini semua huruf di dalam dokumen diubah menjadi huruf kecil, dan karakter yang diambil adalah dari ‘a’ sampai ‘z’ dimana karakter selain itu akan dihilangkan dan dianggap sebagai delimiter (Feldman, 2007).

2. *Tokenizing*

Tahap *Tokenizing* adalah tahap pemotongan *string input* berdasarkan tiap kata yang menyusunnya (Sholehhudin et al., 2018).

3. *Stopword removal* atau *filtering*

Tahap filtering merupakan tahap mengambil kata-kata penting dari hasil token. Bisa menggunakan algoritma *stoplist* (membuang kata yang kurang penting) atau *wordlist* (menyimpan kata penting). *Stoplist / stopword* adalah kata-kata yang tidak deskriptif yang dapat dibuang dalam pendekatan BOW (*bag-of-words*) (Feldman, 2007).

4. *Stemming*

Steming adalah proses membentuk suatu kata menjadi kata dasarnya atau sebuah proses mencari akar kata dari setiap kata hasil *filtering*. Dengan proses *stemming* ini, setiap kata yang miliki imbuhan akan berubah menjadi kata dasar dan dapat lebih mengoptimalkan proses dari *text mining* (Sholehhudin et al., 2018).

2.2 FastText

FastText merupakan jaringan syaraf tiruan dengan tiga komponen utama yaitu lapisan masukan (*word embedding*), lapisan tersembunyi, dan lapisan keluaran (Herwanto et al., 2019). FastText merupakan *library* yang dikeluarkan oleh Facebook yang merupakan pengembangan dari Word2Vec. FastText dan Word2Vec sama-sama dapat dipergunakan untuk word embedding dan juga mendeteksi kesamaan kata secara semantik maupun sintaksis.

FastText memetakan setiap kosakata ke vektor bernilai nyata, dengan kata-kata yang tidak dikenal memiliki ID kosakata khusus. Sebuah dokumen dapat direpresentasikan sebagai rata-rata dari semua vektor ini. Kemudian FastText akan melatih pengklasifikasi multi-kelas entropi maksimum pada vektor dan label keluaran. FastText telah terbukti melatih dengan cepat dan mencapai performa prediksi yang sebanding dengan model penyematan Recurrent Neural Network untuk klasifikasi teks (Joulin et al., 2016).

Terdapat dua jenis arsitektur pada dalam fastText yaitu *Contious Bag of Words* (CBOW) dan *Skip-gram*. Perbedaan dari CBOW dan Skip-gram adalah cara CBOW yang bekerja cenderung memprediksi probabilitas kata sebagai target yang diberikan konteks sebagai input. Sedangkan Skip-gram adalah kebalikannya dari CBOW, yang berarti konteks dapat berupa satu kata atau kelompok kata.

2.3 Multilayer Perceptron

Multilayer Perceptron adalah kelas dari jaringan syaraf *feedforward* yang dibangun oleh grafik asiklik berlapis. Sebuah *Multilayer Perceptron* terdiri setidaknya dari tiga lapisan dan aktivasi non-liner. Lapisan pertama disebut dengan lapisan masukan, lapisan kedua disebut dengan lapisan tersembunyi dan lapisan ketiga disebut lapisan keluaran. Ketiga dari lapisan tersebut terhubung penuh yang berarti setiap node pada hidden layer terhubung ke setiap node pada layer lainnya. Multilayer perceptron dilatih menggunakan *backpropagation*, dimana bobot diperbarui dengan menghitung penuruan gradien sehubungan fungsi error (Nyberg, 2018).

Berikut adalah langkah-langkah algoritma yang dijalankan oleh *Multilayer Perceptron* untuk mendapatkan bobot optimal (Muliantara & Widiartha, 2011):

1. Inisialisasi semua bobot dengan bilangan acak kecil.
2. Jika kondisi penghentian belum dipenuhi, ikuti langkah 2 – 8.
3. Setiap pasang data pelatihan, lakukan langkah 3 – 8.
4. Tiap lapisan masukan menerima sinyal dan meneruskan ke lapisan tersembunyi di atasnya.
5. Hitung semua *output* di lapisan tersembunyi dengan Z_j (j = 1, 2, …,p).

$$z\_net_j = v_{j0} + \sum_{i=1}^{n} x_i v_{ji} \quad \text{…(2.1)}$$

$$z_j = f(z\_net_j) = \frac{1}{1 + e^{-z\_net_j}} \quad \text{…(2.2)}$$

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

$$y\_net_k = w_{k0} + \sum_{i=1}^{p} z_i w_{kj} \quad \text{…(2.3)}$$

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

7. Hitung faktor δ lapisan keluaran berdasarkan kesalahan disetiap lapisan keluaran y_k (k = 1, 2, …, m).

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

$$t_k = target$$

δ_k merupakan unit kesalahan yang akan dipakai dalam perubahan bobot layer di bawahnya. Hitung perubahan bobot W_{kj} dengan lanju pemahaman α.

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

K = 1, 2, …, m; j = 0,1, …, p

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

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

Faktor δ unit tersembunyi.

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

Hitung suku perubahan bobot v_{ij}

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

j = 1, 2, …, p; i = 1, 2, …, n

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

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

(k = 1, 2, …, m; j= 0, 1, …, p)

Perubahan bobot garis yang menuju ke unit tersembunyi yaitu :

$$v_{ij}(baru) = v_{ij}(lama) + \Delta v_{ij} \quad \ldots(2.11)$$

j = 1, 2, …, p; i = 0, 2, …, n

2.4 Evaluasi Klasifikasi

Terdapat beberapa cara untuk mengukur kinerja metode klasifikasi diantaranya dengan menggunakan *Confusion Matrix, Precision, Recall,* dan *F1-Score*. *Confusion Matrix* merupakan sebuah table yang terdiri atas banyaknya baris

data uji diprediksi dengan benar dan tidak benar oleh model klasifikasi, yang dipergunakan untuk menentukan kinerja dari suatu model klasifikasi (Hamel, 2010). Tabel *Confusion Matrix* dapat membantu dalam memperoleh nilai dari perhitungan tersebut karena berisi data prediksi positif dan negative yang dihasilkanoleh system dan data actual yang positif dan negative di dunia nyata (Goutte & Gaussier, 2005).

Berikut adalah bentuk dari table *Confusion Matrix* :

Tabel 2.1 *Confusion Matrix Table*

	Positif (Aktual)	Negatif (Aktual)
Positif (Sistem)	TP	FP
Negatif (Sistem)	FN	TN

Dimana penjelasan *variable* sebagai berikut, TP (*True Positive*) adalah perhitungan dari kelas positif sistem dan actual yang positif. TN (*True Negatif*) adalah perhitungan dari kelas negatif sistem dan aktual yang negatif. FP (*False Positive*) adalah perhitungan dari kelas positif sistem dan aktual yang negatif. FN (*False Negative*) adalah perhitungan dari kelas negatif sistem dan aktual yang positif.

Precision merupakan rasio kategorisasi dokumen yang benar ke dalam kategori dengan jumlah total percobaan klasifikasi. *Recall* merupakan tingkat kemampuan sistem dalam menemukan kembali sebuah informasi yang relevan. (Forman, 2003) Untuk menghindari perbedaan rasio yang cukup tinggi antara nilai *precision* dan *recall*, maka dilakukan penyetaraan nilai dengan *F1-Score* (Ponweiser, 2012). *F1-Score* adalah perhitungan evaluasi dalam informasi *retrieval*

yang mengkombinasikan *recall* dan *precision* (Erwin et al., 2019). Adapun nilai *precision, recall* dan F1-Score yang dapat dihigung menggunakan rumus sebagai berikut

$$Precision = \frac{true\ positive}{true\ positive + false\ positive} \quad \ldots(2.12)$$

$$Recall = \frac{true\ positive}{true\ positive + false\ negative} \quad \ldots(2.13)$$

$$F1 - Score = 2 \times \frac{Precission \times Recall}{Precission + Recall} \quad \ldots(2.14)$$