BAB II

LANDASAN TEORI

2.1 Text Classification

Text Classification atau klasifikasi teks merupakan salah satu bidang dari *Natural Language Processing* yang mengotomasikan pengklasifikasian teks ke satu atau lebih kategori yang tepat berdasarkan isinya dengan membangun model menggunakan data latih (Mowafy et al., 2018). Klasifikasi teks diterapkan dalam berbagai konteks, mulai dari pengindeksan dokumen berdasarkan kosa kata yang terkontrol, penyaringan dokumen, pembuatan *metadata* otomatis, dan berbagai aplikasi lainnya yang membutuhkan organisir dokumen (Sebastiani, 2001). Ada beberapa strategi umum dalam penggunaan klasifikasi teks, yaitu *text preprocessing*, *feature extraction*, *modeling* menggunakan teknik pembelajaran mesin yang sesuai, serta *training* dan *testing* pada *classifier* (Dalal dan Zaveri, 2011).

2.2 Text Preprocessing

Komentar pengguna Twitter mengandung banyak kata dan kalimat yang tidak memiliki format khusus dan dapat diekspresikan dengan bebas tanpa adanya pembatasan. Oleh sebab itu, sebelum melakukan *text classification*, data yang tidak diperlukan dan berlebihan harus dihilangkan. Beberapa tahap *preprocessing* data komentar pengguna Twitter yang dilakukan dalam penelitian ini adalah *case folding*, *tokenizing*, *filtering*, dan *stemming*.

2.2.1 Case Folding

Case folding adalah proses mengubah semua huruf dalam dokumen menjadi huruf kecil (Triawati, 2009). Hal ini dilakukan karena tidak adanya format atau pembatasan yang mengharuskan seluruh komentar Twitter konsisten dalam penggunaan huruf kapital. Contoh dari proses *case folding* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Contoh Proses Case Folding
(Damanik, 2014)

Sebelum Case Folding	**Setelah Case Folding**
Manajemen	manajemen
pengetahuan	pengetahuan
adalah	adalah
sebuah	sebuah
konsep	konsep
baru	baru
di	di
dunia	dunia
bisnis	bisnis

2.2.2 Tokenizing

Tokenizing adalah proses pemecahan suatu teks menjadi kata, frasa, simbol, atau elemen bermakna lainnya yang disebut sebagai token (Gurusamy & Kannan, 2014). Proses ini bertujuan untuk melakukan eksplorasi pada setiap kata dalam

sebuah kalimat untuk mengidentifikasi kata kunci yang bermakna dalam kalimat tersebut. Contoh dari proses *tokenizing* dapat dilihat pada Tabel 2.2.

Tabel 2.2 Contoh Proses Tokenizing
(Damanik, 2014)

Sebelum Tokenizing	**Setelah Tokenizing**
Manajemen pengetahuan adalah sebuah konsep baru di dunia bisnis.	Manajemen
	pengetahuan
	adalah
	sebuah
	konsep
	baru
	di
	dunia
	bisnis

2.2.3 Filtering

Filtering adalah tahap mengambil kata-kata penting dari hasil token dengan melakukan penghilangan *stopwords* (Putri, 2017). *Stopwords* adalah kata yang bukan merupakan kata unik dalam suatu artikel atau kata-kata umum yang biasanya selalu ada dalam suatu artikel (Mooney, 2006). Contoh dari proses *filtering* dapat dilihat pada Tabel 2.3.

Tabel 2.3 Contoh Proses Filtering
(Damanik, 2014)

Sebelum Filtering	**Setelah Filtering**
manajemen	manajemen
pengetahuan	pengetahuan
adalah	
sebuah	
konsep	konsep
baru	baru
di	
dunia	dunia

2.2.4 Stemming

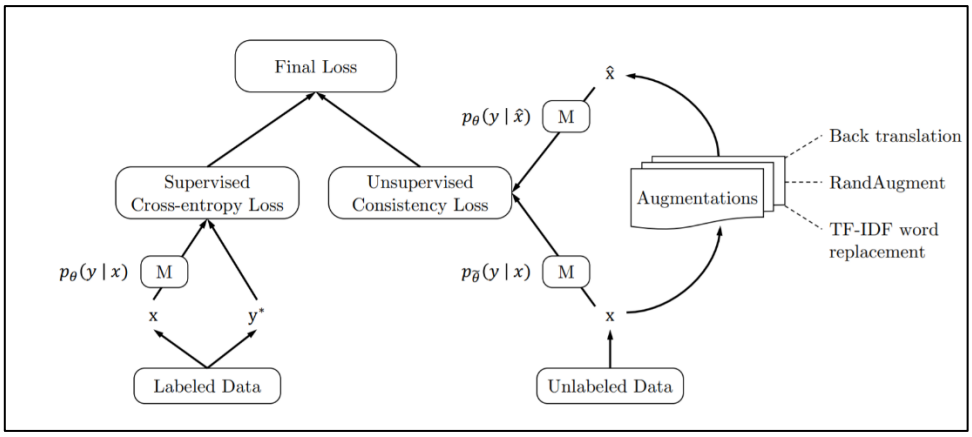
Stemming adalah proses pemetaan variansi morfologikal kata dalam kata dasar atau kata umumnya (Adhitia dan Purwarianti, 2012). Proses *stemming* pada teks Bahasa Indonesia memerlukan proses penghilangan *prefix* (awalan), seperti "ter-", "pe-", "se-", "ke-", "ber", dan lain sebagainya, serta *suffix* (akhiran), seperti "-kan", "-i", "-nya", dan lain sebagainya. Contoh dari proses *stemming* dapat dilihat pada Tabel 2.4.

Tabel 2.4 Contoh Proses Stemming
(Vijrayani et al., 2015)

Sebelum Stemming	**Setelah Stemming**
Connected	*Connect*
Connecting	
Connections	

2.3 Unsupervised Data Augmentation

Unsupervised Data Augmentation (UDA) merupakan metode *semi-supervised learning* yang menggunakan data berlabel dan data tidak berlabel dalam proses pelatihannya. Berbeda dengan metode *semi-supervised* lainnya, UDA mengurangi kebutuhan akan data berlabel dan lebih baik dalam memanfaatkan data tidak berlabel (Xie et al., 2019). Untuk data berlabel, UDA menghitung *loss function* dengan menggunakan metode standar *supervised learning* dalam melatih model yaitu *Cross-entropy loss*. Untuk data tidak berlabel, *consistency training* diterapkan untuk memaksa prediksi yang dihasilkan terhadap data tidak berlabel dapat serupa dengan data dari hasil augmentasi. Kedua jenis data tidak berlabel tersebut digunakan untuk membangun dua hasil prediksi model yang kemudian digunakan untuk menghitung *consistency loss*. Kemudian UDA menghitung *final loss* dengan mengoptimalkan *supervised loss* pada data berlabel dan *consistency loss* pada data tidak berlabel. Secara garis besar, cara kerja dari UDA dapat dilihat pada Gambar 2.1.



Gambar 2.1 Cara Kerja Unsupervised Data Augmentation
(Xie et al., 2019)

Dengan meminimalkan *consistency loss*, UDA memungkinkan informasi label menyebar dengan lancar dari contoh berlabel ke contoh tidak berlabel. Secara intuitif, UDA dapat dianggap sebagai proses iteratif implisit. Pertama, model bergantung pada sejumlah kecil contoh berlabel untuk membuat prediksi yang benar pada beberapa contoh tidak berlabel. Hal tersebut menyebabkan informasi label menyebar ke data dari hasil augmentasi melalui *consistency loss*. Seiring berjalannya iterasi, semakin banyak contoh tidak berlabel yang dapat diprediksi dengan benar, dimana hal tersebut juga meningkatkan generalisasi dari model. Metode augmentasi yang digunakan oleh UDA tergantung pada pekerjaan yang dijalankan. Metode-metode augmentasi tersebut berupa RandAugment untuk *image classification* dan *back translation* serta Term Frequency-Inverse Document Frequency (TF-IDF) *word replacement* untuk *text classification*. *Back translation* merupakan metode augmentasi *sentence-level*, sedangkan TF-IDF *word replacement* merupakan metode augmentasi *word-level*. Dalam penggunaannya, UDA bekerja sebagai bagian dari BERT untuk melakukan *text classification*. Oleh sebab itu, UDA juga menggunakan BERT sebagai *feature extraction method* untuk merepresentasikan teks ke dalam bentuk numerik.

Menurut Devlin et al. (2018), BERT merupakan metode representasi bahasa pra-pelatihan, yang berarti bahwa model dilatih dengan menggunakan korpus teks besar yang digunakan untuk *Natural Language Processing* (NLP). BERT merupakan metode *unsupervised* pertama yang menggunakan sistem *bidirectional* untuk pra-pelatihan NLP. Sistem *bidirectional* tersebut dapat merepresentasikan suatu teks dalam dua arah, sehingga menjadi kelebihan BERT jika dibandingkan dengan sistem *unidirectional*. Sebagai contoh terdapat kalimat “I made a bank deposit”, representasi

unidirectional dari “bank” hanya bergantung pada konteks kirinya saja yaitu “I made a”, bukan “deposit”. Sedangkan representasi *bidirectional* dari “bank” bergantung pada konteks kiri dan konteks kanan yaitu “I made a” dan “deposit” (Devlin et al., 2018).

2.4 Metrik Evaluasi

Menurun Hossin dan Sulaiman (2015), metrik evaluasi merupakan *evaluator* yang dapat digunakan untuk pemilihan model atau mengevaluasi kemampuan generalisasi dari model yang dilatih. Dalam masalah *binary-classification* dimana hanya terdapat dua kelas berlawanan yang diprediksi (Canbek et al., 2017), solusi terbaik untuk mengevaluasi model dapat ditentukan berdasarkan *Confusion Matrix* (Hossin dan Sulaiman, 2015). Menurut Ting (2017), *Confusion Matrix* merupakan matriks dua dimensi, dimana satu dimensi diindeks oleh kelas sebenarnya dari suatu objek dan dimensi lainnya diindeks oleh kelas yang ditentukan model. Struktur dari *Confusion Matrix* dapat dilihat pada Tabel 2.5.

Tabel 2.5 Struktur Confusion Matrix
(Hossin dan Sulaiman, 2015)

Aktual \ Prediksi	**Positif**	**Negatif**
Positif	TP	FN
Negatif	FP	TN

TP (*True Positive*), TN (*True Negative*), FP (*False Positive*), dan FN (*False Negative*) merupakan nilai dari *Confusion Matrix*, yang kemudian digunakan untuk

mengukur performa dari model. Menurut Hossin dan Sulaiman (2015), nilai *accuracy* digunakan untuk mengukur ketepatan prediksi benar yang dihasilkan terhadap total jumlah contoh yang dievaluasi. Menurut Hossin dan Sulaiman (2015), formula perhitungan nilai *accuracy* dapat dilihat pada Persamaan 2.1.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad \ldots(2.1)$$

Akan tetapi, nilai *accuracy* memiliki beberapa kelemahan seperti kurang informatif dan bias, sehingga digunakan metrik pengukuran lain untuk membantu dalam pengukuran performa model, yaitu nilai *precision*, *recall*, dan *F1 score*. Nilai *precision* digunakan untuk mengukur pola positif yang diprediksi dengan benar (*True Positive*) dari total pola prediksi dalam kelas positif. Menurut Hossin dan Sulaiman (2015), formula perhitungan nilai *precision* dapat dilihat pada Persamaan 2.2.

$$Precision = \frac{TP}{TP + FP} \quad \ldots(2.2)$$

Nilai *recall* digunakan untuk mengukur pecahan dari pola positif yang diklasifikasikan dengan benar. Menurut Hossin dan Sulaiman (2015), formula dari perhitungan nilai *recall* dapat dilihat pada Persamaan 2.3.

$$Recall = \frac{TP}{TP + FN} \quad \ldots(2.3)$$

Nilai *F1 score* merupakan rata-rata antara nilai *precision* dan *recall*. Menurut Hossin dan Sulaiman (2015), formula dari perhitungan nilai *F1 score* dapat dilihat pada Persamaan 2.4.

$$F1\ Score = \frac{2\ x\ Precision\ x\ Recall}{Precision + Recall} \quad \ldots(2.4)$$