BAB 2

LANDASAN TEORI

2.1 Definisi Penyandang Disabilitas

Menurut International Labour Organization (2014), penyandang disabilitas adalah seseorang yang mengalami kelainan fisik, indera, intelektual, maupun psikososial yang dapat memengaruhi kemampuan orang tersebut dalam menjalani kegiatan sehari-harinya. Menurut The United States Department of Justice (2016), disabilitas adalah kecacatan substansial baik dalam bentuk fisik maupun mental yang membatasi aktivitas hidup suatu individu, memiliki riwayat kecacatan, atau dianggap memiliki kecacatan. Menurut Chhabra (2016) difabel / *diffable* (differently abled) adalah seseorang yang mengalami kecacatan yang menyebabkan fungsional, batasan dalam melakukan aktivitas, atau kecacatan sosial.

Penyebutan pada penyandang disabilitas sendiri dapat diklasifikasikan berdasarkan tipe dan jenis disabilitas dengan mengacu pada kondisi yang dialami oleh penyandang disabilitas itu sendiri (Republik Indonesia, 1997; Handayani, 2017; Khairunisa Rani, Rafikayati and Jauhari, 2018; Purnamasari, P,.& Soendari, 2018; Puspitawati and Darmadha, 2019). Klasifikasi tipe dan jenis penyandang disabilitas yang dimaksud adalah sebagai berikut.

1) Disabilitas fisik

   Disabilitas fisik adalah kecacatan yang mengakibatkan gangguan pada fungsi tubuh baik itu secara gerak tubuh, penglihatan, pendengaran maupun

kemampuan bicara. Tipe disabilitas yang termasuk disabilitas fisik adalah sebagai berikut.

a) Tipe A (tunanetra)

Tunanetra merupakan gangguan pada organ penglihatan sehingga mengakibatkan ketidakmampuan melihat pada penyandangnya.

b) Tipe B (tunarungu)

Tunarungu merupakan gangguan pada organ pendengaran sehingga mengakibatkan ketidakmampuan mendengar pada penyandangnya.

c) Tipe C (tunawicara)

Tunawicara merupakan gangguan yang mengakibatkan ketidakmampuan berbicara pada penyandangnya.

d) Tipe D (tunadaksa)

Tunadaksa merupakan gangguan pada anggota tubuh sehingga mengakibatkan menurunnya kemampuan gerak pada penyandangnya.

e) Tipe E1 (tunalaras)

Tunalaras merupakan gangguan yang mengakibatkan menurunnya kemampuan bersosialisasi atau berinteraksi sosial pada penyandangnya. Tunalaras pada tipe E1 biasanya mengalami cacat pada suara dan nada berbicara

2) Disabilitas mental

Disabilitas mental adalah kelainan yang mempengaruhi kondisi mental atau tingkah laku penyandangnya. Disabilitas mental mencakup kelainan

bawaan lahir maupun akibat dari penyakit. Tipe disabilitas yang termasuk disabilitas mental adalah sebagai berikut.

a) Tipe E2 (tunalaras)

Tunalaras merupakan gangguan yang mengakibatkan menurunnya kemampuan bersosialisasi atau berinteraksi sosial pada penyandangnya. Penyandang tunalaras tipe E2 mengalami gangguan emosional dan penyimpangan tingkah laku.

b) Tipe F (tunagrahita)

Tunagrahita merupakan gangguan yang mengakibatkan rendahnya tingkat kecerdasan terutama pada bidang akademik pada penyandangnya.

3) Disabilitas ganda atau disabilitas fisik dan mental

Disabilitas ganda adalah keadaan seseorang yang menyandang dua jenis kecacatan sekaligus. Tipe disabilitas yang termasuk disabilitas ganda adalah sebagai berikut.

a) Tipe G (tunaganda)

Tunaganda adalah keadaan dimana penyandangnya mengalami dua jenis gangguan sekaligus.

2.2 Definisi dan Konsep Dasar Peringkasan Teks Otomatis

Peringkasan Teks Otomatis adalah suatu kegiatan yang berjalan secara otomatis dan dapat menghasilkan teks ringkasan yang padat tetapi masih mampu mempertahankan makna utama juga makna keseluruhan dari teks yang ingin diringkas serta menggunakan bahasa yang koheren. (Allahyari *et al.*, 2017).

Peringkasan teks otomatis dapat menguntungkan pembaca dalam menghemat waktu dan tenaga. Karena pembaca mampu mendapatkan poin utama dari sebuah teks tanpa perlu membacanya secara menyeluruh (El-Kassas *et al.*, 2021).

Peringkasan Teks secara umum dibedakan menjadi 2 jenis berdasarkan pendekatan yang dilakukan adalah sebagai berikut.

1) Pendekatan ekstraktif

   Pendekatan secara ekstraktif bekerja dengan cara mengidentifikasi bagian-bagian penting dari teks yang ingin diringkas dan menyusunnya secara verbatim untuk menghasilkan teks ringkasan (Allahyari *et al.*, 2017). Pendekatan secara ekstraktif memiliki kelebihan pada sisi akurasi, waktu dan *simplicity* daripada pendekatan secara abstraktif (El-Kassas *et al.*, 2021). Pendekatan secara ekstraktif merupakan pendekatan yang dilakukan pada penelitian ini.

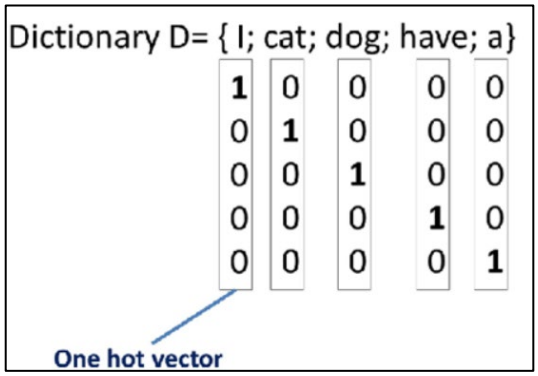
2) Pendekatan abstraktif

   Peringkasan abstraktif bekerja dengan cara memeriksa dan menafsirkan teks yang ingin diringkas menggunakan *advanced natural language techniques* dan menciptakan teks baru yang dapat menyampaikan informasi penting dalam bentuk yang lebih ringkas dari teks asal (Allahyari *et al.*, 2017).

2.3 One-hot Encoding

One-hot Encoding merupakan salah satu cara merepresentasikan kata dalam bentuk *vector* dan bahkan merupakan metode *encoding* yang terbilang simpel dan banyak digunakan (Potdar, S. and D., 2017; Cerda, Varoquaux and Kégl, 2018). Jain (2020b) melakukan penelitian yang berjudul Fine-Tuning Textrank for Legal

Document Summarization: A Bayesian Optimization Based Approach juga menggunakan *One-hot vector based TextRank* sebagai *baseline model*snya. One-hot encoding bekerja dengan cara memberikan nilai berbeda pada setiap kata lalu mengubah kata ke-d menjadi nilai 0 atau 1 ke dalam vector dengan dimensi sebesar n (jumlah kata dalam teks). Nilai 0 mengindikasikan ketidakhadiran dan nilai 1 mengindikasikan kehadiran variabel ke-d (Potdar, S. and D., 2017; Ul Haq *et al.*, 2019). Contoh representasi kata dalam *One-hot Vector* dapat dilihat pada gambar 2.1.



Gambar 2.1 Representasi Kata Dalam *One-Hot Vector*

Sumber: (Nguyen *et al.*, 2016)

2.4 Term Frequency - Inverse Document Frequency (TF-IDF)

TF-IDF merupakan salah satu metode yang digunakan dalam merepresentasikan dokumen ke dalam *multi-dimensional vector* dan termasuk metode *term-weighting* yang paling banyak digunakan (Beel, Joeran and Langer, Stefan and Gipp, 2017; Marwah and Beel, 2020; Marcińczuk Michałand Gniewkowski, Walkowiak and Kedkowski, 2021). TF-IDF bekerja dengan cara melakukan perbandingan antara seberapa banyak kemunculan sebuah kata dalam

dokumen (*term frequency*) dengan *inverse document frequency* dari kata tersebut (*inverse document frequency*) (Qaiser and Ali, 2018).

Perhitungan TF-IDF dapat dilihat pada persamaan berikut.

$$TF - IDF(t,d) = TF(t,d) \times IDF(t) \tag{2.1}$$

Keterangan:

$TF - IDF(t,d)$ = nilai TF-IDF pada *term* ke-t, dokumen ke-d

$TF(t,d)$ = nilai *term frequency* pada *term* ke-t, dokumen ke-d

$IDF(t)$ = nilai *inverse document* frequency pada *term* ke-t

Perhitungan *term frequency* (tf) dapat dilihat pada persamaan berikut.

$$TF(t,d) = \frac{Term\ t\ frequency\ in\ document\ d}{total\ words\ in\ document\ d} \tag{2.2}$$

Rumus *inverse document frequency* (idf) dapat dilihat pada persamaan berikut.

$$IDF(t) = \log_{10}\left(\frac{n}{df(t)}\right) \tag{2.3}$$

Keterangan:

n = jumlah kalimat pada dokumen

$df(t)$ = nilai *document frequceny* pada *term* ke-t

Perhitungan *inverse document frequency* (idf) pada library Scikit-learn sedikit berbeda. Pada function TfidfVectorizer jika parameter *smooth_idf* bernilai *false,* dilakukan pertambahan nilai 1 pada hasil perhitungan logaritma sehingga *term* yang muncul pada seluruh dokumen tidak diabaikan sedangkan jika parameter *smooth_idf* bernilai *true* pertambahan nilai 1 pada pembilang, penyebut dan hasil

perhitungan logaritma demi menghindari pembagian nilai 0 pada perhitungan *term* yang muncul pada seluruh dokumen.

Perhitungan *idf* saat parameter *smooth_idf* bernilai *false* dapat dilihat pada persamaan berikut

$$IDF(t) = \log_{10}\left(\frac{n}{df(t)}\right) + 1 \tag{2.4}$$

Perhitungan *idf* saat parameter *smooth_idf* bernilai *true* dapat dilihat pada persamaan berikut

$$IDF(t) = \log_{10}\left(\frac{n+1}{df(t)+1}\right) + 1 \tag{2.5}$$

2.5 FastText

FastText adalah *open-source, free, lightweight library* yang dapat melakukan *text representation* dan *text classifier* (Bojanowski *et al.*, 2017; Joulin *et al.*, 2017; Grave *et al.*, 2019). FastText mampu menghasilkan vektor dari kata yang tidak diketahui karena FastText mempertimbangkan suku kata dari sebuah kata sedangkan model lain biasanya mengabaikan atau memberikan *random vector* (Bojanowski *et al.*, 2017; Ruseti, 2019). FastText juga menyediakan *pre-trained* CBOW model yang dilatih menggunakan Common Crawl dan Wikipedia dalam 157 bahasa berbeda dan Bahasa Indonesia adalah salah satunya (Joulin *et al.*, 2017; Grave *et al.*, 2019; Young and Rusli, 2019).

FastText menyediakan 2 model dalam komputasi word representation yaitu skipgram dan CBOW (continuous-bag-of-words). Skipgram mampu belajar dari *training* data yang terbilang kecil dan mampu merepresentasikan kata yang jarang muncul dengan lebih baik sedangkan CBOW memiliki waktu *training* yang lebih

singkat dan dapat merepresentasikan kata yang sering muncul dengan lebih baik (Giatsoglou *et al.*, 2017). Bansal (2018) melakukan penelitian dengan mengkombinasikan model skipgram dan CBOW pada algoritma SVM, Naïve Bayes, Logistic Regression dan Random Forest dan berkesimpulan bahwa kombinasi model CBOW dan Random Forest mampu mendapatkan akurasi tetinggi. Skipgram model bekerja lebih baik daripada CBOW dalam hal *subword information* (Bojanowski *et al.*, 2017).

2.6 Konsep Dasar Metode Textrank

TextRank merupakan salah satu metode dalam peringkasan teks otomatis yang menggunakan pendekatan ekstraktif dan termasuk ke dalam metode dengan *graph-based model* (Setiadi B, 2017; Akhtar, Beg and Javed, 2019; Gunawan, Harahap and Fadillah Rahmat, 2019; Jain, Borah and Biswas, 2020a). Metode *TextRank* melakukan perhitungan nilai *similarity* pada setiap kalimat dan menyimpannya ke dalam bentuk *matrix similarity,* lalu *similarity matrix* tersebut diubah menjadi sebuah graf. Graf pada metode *TextRank* merepresentasikan teks yang ingin diringkas, kalimat dalam teks tersebut akan direpresentasikan dalam bentuk *node,* dan *similarity* antar kalimat direpresentasikan dalam bentuk *edge.* Pada Langkah terakhir, *TextRank* akan melakukan pemeringkatan berdasarkan graf yang sudah dibuat sebelumnya dan kalimat atau *node* dengan *rangking* tertinggi akan diikutsertakan ke dalam teks ringkasan (Barrios *et al.*, 2016; Ashari and Riasetiawan, 2017; Xiong *et al.*, 2018).

TextRank menghitung nilai *similarity* antara dua buah kalimat (*node*) menggunakan hasil pembagian antara jumlah token leksikal milik kedua kalimat

dengan panjang masing-masing kalimat. Perhitungan *similarity* pada metode *TextRank* juga dapat mengurangi kalimat yang terlalu panjang pada hasil akhir teks ringkasan (Barrios *et al.*, 2016). *Cosine similarity* merupakan salah satu metode yang bisa digunakan untuk melakukan perhitungan *similarity* antara 2 kalimat (Farouk, 2019).

Pe*rangking*an pada metode *TextRank* dihitung menggunakan persamaan berikut.

$$WS(V_i) = (1-d) + d \times \sum_{V_i \epsilon \text{ In}(V_i)} \frac{\omega_{ij}}{\sum_{V_i \epsilon \text{ out}(V_i)} \omega_{jk}} WS(V_j) \tag{2.6}$$

Keterangan:

$WS(V_i)$ = *weight sentences* atau bobot kalimat V_i (*TextRank score*)

V_i = *vertex* / *node* / kalimat i

$WS(V_j)$ = *weight sentences* atau bobot kalimat V_j (*TextRank score*)

V_j = *vertex* / *node* / kalimat j

d = *damping coefficient* (bernilai antara 0 sampai 1)

ω_{ij} = bobot *edge* antara kalimat i dan j

ω_{jk} = bobot *edge* antara kalimat j dan k

2.7 Konsep Dasar Metode Maximum Marginal Relevance

Metode MMR akan melakukan pemeringkatan ulang dari hasil ringkasan sebelumnya yang mengimplementasikan metode *TextRank*. Pemeringkatan ulang dilakukan karena metode MMR dapat mengurangi terjadinya redundansi kalimat atau beberapa kalimat memiliki makna yang hampir sama pada hasil akhir teks ringkasan (Khan *et al.*, 2018; Gunawan, Harahap and Fadillah Rahmat, 2019; Musyaffanto, Budi Herwanto and Riasetiawan, 2019; Mao *et al.*, 2020).

Metode MMR akan melakukan perhitungan nilai *cosine similarity* pada sebuah kalimat dan membandingkannya dengan kalimat lain yang terdapat pada teks sebelum diringkas (Gunawan, Harahap and Fadillah Rahmat, 2019). Pada penelitian ini, kalimat yang dibandingkan berasal dari teks ringkasan hasil implementasi metode *TextRank* sebelumnya, Setelah perhitungan nilai *cosine similarity*, nilai tersebut akan dihitung kembali untuk mendapatkan nilai MMR, dan kalimat dengan nilai MMR terendah akan dihapus atau tidak diikutsertakan dari teks ringkasan akhir.

Pada metode MMR juga terdapat sebuah bobot parameter untuk mengatur nilai tingkat relevansi kalimat dalam ringkasan yang disebut dengan *lambda* (λ). Nilai *lambda* berkisar antara 0 (nol) hingga 1 (satu). Saat nilai *lambda* mendekati 0 (nol) maka metode Maximal Marginal Relevance akan memilih kalimat dengan makna yang lebih beragam (*diverse*) untuk diikutsertakan dalam ringkasan sedangkan saat nilai *lambda* mendekati 1 (satu) maka metode Maximal Marginal Relevance akan memilih kalimat dengan makna yang lebih seragam (*accurate*) untuk diikutsertakan dalam ringkasan (Gunawan, Harahap and Fadillah Rahmat, 2019).

Perhitungan nilai MMR dilakukan secara terus menerus hingga panjang teks ringkasan akhir mencapai tingkat kompresi yang diinginkan. Perhitungan nilai MMR dilakukan menggunakan persamaan berikut.

$$MMR(S_i) = \lambda \times Sim_1(S_i, Q) - (1 - \lambda) \times \max\left(Sim_2(S_i, S_j)\right) \quad (2.7)$$

Keterangan:

λ = Bobot parameter untuk mengatur tingkat relevansi (0 – 1)

S_i = Vektor bobot kata yang menjadi kandidat

S_j = Vektor bobot kata yang tidak menjadi kandidat

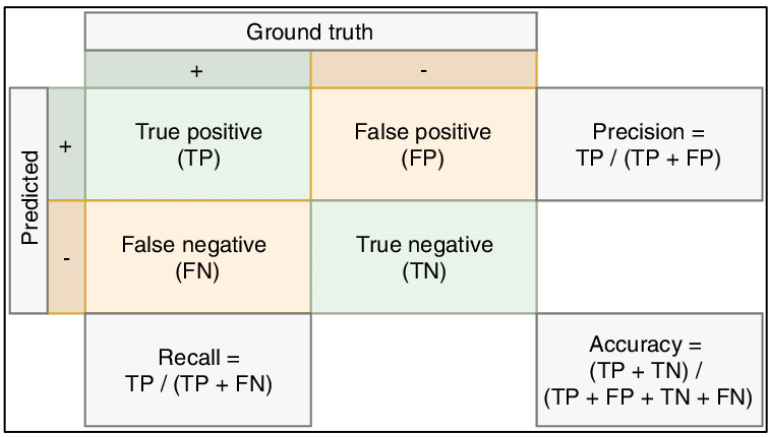
Q = Vektor bobot kata dari *query user*

Sim_1 = nilai *similarity* antara *query* dengan kalimat lainnya

Sim_2 = nilai *similarity* antara kalimat dengan kalimat lain dalam ringkasan

2.8 Confusion Matrix

Confusion matrix merupakan alat evaluasi yang digunakan dalam *machine learning. Confusion Matrix* merupakan matriks 2 x 2 dan kolom pada *confusion matrix* merepresentasikan hasil dari kelas prediksi dan baris pada *confusion matrix* merepresentasikan hasil dari kelas yang sebenarnya (Xu, Zhang and Miao, 2020). Confusion matrix digunakan untuk menghitung *precision* dan *recall*. *Precision* dan *recall* nantinya akan digunakan untuk menghitung F1-*score*. F1-*score*, *Precision* dan *recall* nantinya akan digunakan untuk mengevaluasi hasil akhir teks ringkasan. Tabel *confusion matrix* dapat dilihat pada Gambar 2.2.



Gambar 2.2 Tabel *Confusion Matrix*

Sumber: (Jeppesen *et al.*, 2019)

Precision mengukur kemampuan suatu algoritma dalam melakukan prediksi.

Perhitungan *precision* dapat dilihat pada persamaan berikut.

$$precision = \frac{overlapping\ sentence}{total\ sentence\ on\ system\ summary} \tag{2.8}$$

Recall mengukur keefektifan suatu algoritma pada satu sampel kelas.

Perhitungan *recall* dapat dilihat pada persamaan berikut.

$$recall = \frac{overlapping\ sentence}{total\ sentence\ on\ human\ summary} \tag{2.9}$$

F_1-*score* merupakan *harmonic mean* dari *precision* dan *recall.*

Perhitungan F_1-*score* dapat dilihat pada persamaan berikut

$$F_1\ Score = \frac{2\ \times precision\ \times recall}{precision + recall} \tag{2.10}$$

2.9 Cosine Similarity

Cosine similarity merupakan salah satu dari teknik evaluasi ringkasan dengan menggunakan *content-based measures* (Steinberger and Ježek, 2009; Neha *et al.*, 2018; Ruseti *et al.*, 2018; Schrimpf, 2018; Ruseti, 2019). Perhitungan *cosine similarity* merupakan teknik yang sudah banyak digunakan dalam menghitung *similarity* antara 2 teks yang sudah direpresentasikan ke dalam bentuk vektor (Barrios *et al.*, 2016; Allahyari *et al.*, 2017; Xiong *et al.*, 2018; Khan *et al.*, 2018; Li *et al.*, 2019; Musyaffanto, Budi Herwanto and Riasetiawan, 2019; Ullah and Al Islam, 2019; Vijaya and Reddy, 2019; Gunawan, Harahap and Fadillah Rahmat, 2019; Ahmadi, 2020; Mao *et al.*, 2020; Marwah and Beel, 2020; Ramón-Hernández

et al., 2020; Upasani *et al.*, 2020; Bordoloi *et al.*, 2020; Ferré *et al.*, 2020; Jain, Borah and Biswas, 2020a; Lebanoff, Song and Liu, 2020; El-Kassas *et al.*, 2021).

Perhitungan *cosine similarity* dapat dilihat pada persamaan berikut.

$$\cos(X, Y) = \frac{\sum_{i=1}^{n} X_i \times Y_i}{\sqrt{\sum_{i=1}^{n} X_i^2} \times \sqrt{\sum_{i=1}^{n} Y_i^2}} \quad (2.11)$$

Keterangan:

X = Representasi ringkasan sistem dalam bentuk vector

Y = Representasi teks berita (*reference document*) dalam bentuk vector