2. LANDASAN TEORI

2.1 Tinjauan Pustaka

Subbab mengenai teori umum menjelaskan teori yang ada dalam penelitian kali ini secara umum yang dapat mendukung pemahaman dasar dari konsep dasar dalam penelitian ini.

2.1.1 News Summarization

Ringkasan merupakan suatu teks singkat yang dihasilkan dari kumpulan teks panjang namun tetap menyimpan informasi penting dari teks asalnya (Joshi et al., 2019). Informasi yang menunjukkan ide utama dapat diberikan penilaian dari seberapa tersambungnya informasi dengan tema dan judul berita serta penilaian dari pembaca. Lalu, untuk menentukan informasi penting maka lebih akurat bila dipilih oleh manusia. Karena itulah, perlu dasar untuk melakukan pelatihan terhadap ringkasan yang dibuat oleh manusia serta sudah dipercaya sebagai ringkasan yang baik dengan mengandung informasi penting dari dokumen.

Berdasarkan dari hasilnya, metode untuk peringkasan otomatis dapat dibedakan menjadi dua kategori yaitu abstraktif dan ekstraktif. Metode abstraktif memiliki tujuan untuk melakukan *paraphrase* dari informasi yang dipilih pada dokumen yang akan diringkas. Kemudian, untuk metode ekstraktif memiliki tujuan untuk membuat ringkasan dengan memilih kalimat maupun kata-kata penting dari dokumen yang akan diringkas. Dengan adanya ringkasan dapat mempersingkat waktu untuk menemukan informasi penting dari bacaan (El-Kassas et al., 2020).

2.1.2 Indosum

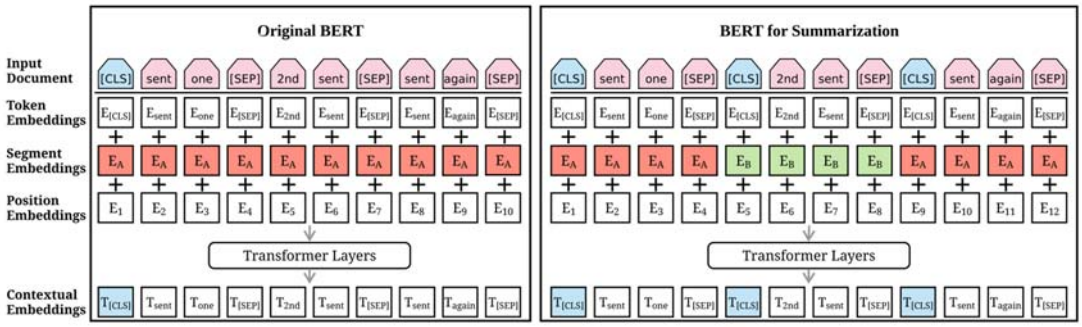
Indosum merupakan *dataset* berita berbahasa Indonesia yang merupakan hasil penelitian yang dilakukan oleh Kurniawan dan Louvan. *Dataset* pada penelitian dapat dijadikan sebagai tolok ukur dalam pembuatan ringkasan otomatis berbahasa Indonesia. *Dataset* ini berbentuk json. Selain itu, dataset ini juga berisi 18.774 artikel berita yang diambil dari berbagai portal berita seperti CNN Indonesia, Kumparan, dan portal berita lainnya. Dalam *dataset* ini terdapat *category* yang menunjukkan kategori berita, *gold label* yang merupakan label *gold summarization* untuk peringkasan ekstraktif, *id* unik tiap berita dari berita yang bersangkutan, *source* yang berisi sumber berita, *source url* yang berisi *url* dari berita, *paragraph* yang berisi berita yang sudah dilakukan pemecahan ke dalam *array*, dan *summary* yang berisi ringkasan berita yang dibuat oleh manusia dan bersifat abstraktif (Kurniawan & Louvan, 2018).

Berikut ini merupakan salah satu contoh isi data dari *dataset*:

{
"category": "olahraga",
"gold_labels": [[true], [false], … ,[false]],
"id": "1503595411-evaluasi-indonesia-untuk-hadapi-malaysia",
"paragraphs": [[["JUARA.NET", ",", "KUALA", "LUMPUR", "-", "Tim", "bulu", "tangkis", "putra", "Indonesia", "akan", "berhadapan", "dengan", "Malaysia", "pafa", "final", "beregu", "SEA", "Games", "2017", "di", "Axiata", "Arena", ",", "Bukit", "Jalil", ",", "Malaysia", ",", "Kamis", "(", "24", "/", "8", "/", "2017", ")", "."]], … ,[["\"", "Saya", "rasa", "pertandingan", "akan", "ramai", ",", "baik", "di", "tunggal", "maupun", "ganda", ",", "\"", "ujar", "Indra", "."]], [["Final", "beregu", "putra", "dijadwalkan", "pada", "Kamis", "(", "24", "/", "8", "/", "2017", ")", "mulai", "pukul", "12.45", "waktu", "setempat", "atau", "13.45", "WIB", "."]]],
"source":"juara.net",
"source_url":"http://juara.bolasport.com/read/raket/bulu-tangkis/1820702-evaluasi-indonesia-untuk-hadapi-malaysia-pada-final-beregu-putra-sea-games-2017",
"summary": [["Tim", "bulu", "tangkis", "putra", "Indonesia", "akan", "berhadapan", "dengan", "Malaysia", "pafa", "final", "beregu", "SEA", "Games", "2017", "di", "Axiata", "Arena", ",", "Bukit", "Jalil", ",", "Malaysia", ",", "Kamis", "(", "24", "/", "8", "/", "2017", ")", "."], …, ["Pemain", "juga", "sebaiknya", "tak", "terpengaruh", "apa", "pun", "dan", "fokus", "pada", "pertandingan", "."]]
}

2.1.3 Bidirectional Encoder Representations from Transformers (BERT)

BERT merupakan arsitektur *pre-trained model multi-layer bidirectional transformer encoder* untuk keperluan *Natural Language Processing*. Model ini dapat digunakan untuk kebutuhan seperti *text classification*, *question answering*, dan lain-lain (Clark et al., 2019). Mengenai struktur dari BERT dapat dilihat pada Gambar 2.1.

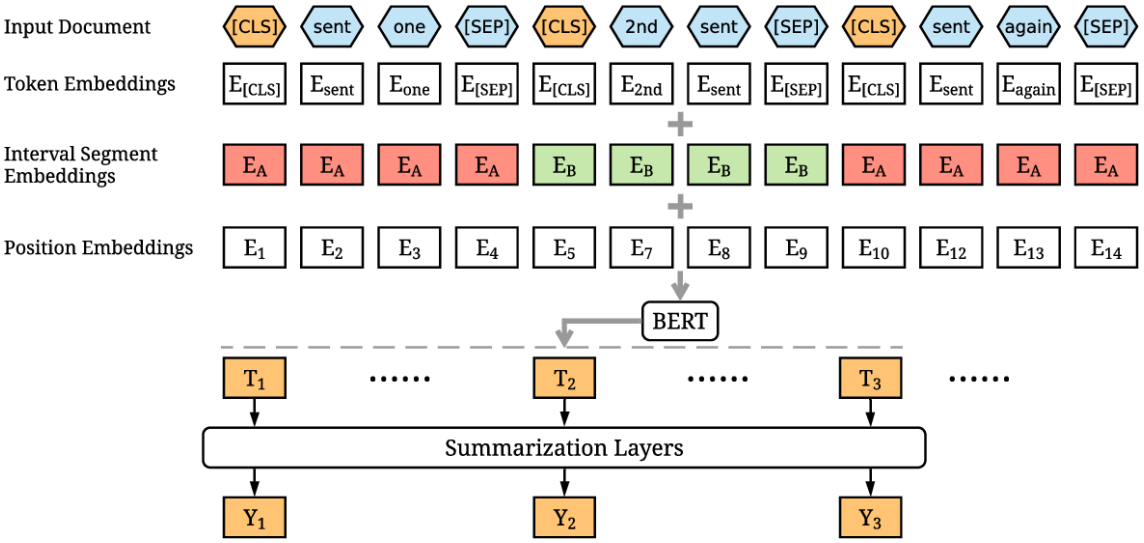


Gambar 2.1 Perbandingan struktur BERT

Sumber: Liu, Y., & Lapata, M. (2019). Text summarization with pretrained encoders. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 3730–3740. https://doi.org/10.18653/v1/d19-1387

Pada BERT terdapat dua tahapan yang dilakukan yaitu *pre-training* dan *fine-tuning*. Selama *pre-training*, akan dilakukan dengan *Masked Language Models* (MLM) dimana secara *random* melakukan *mask* terhadap beberapa *token* dari input kalimat dan tujuannya untuk memprediksi kata yang di-*masked* berdasarkan pada konteks kalimat tersebut. Pada BERT, input dari suatu teks kalimat akan diubah menjadi *token sequence*. Dimana terdapat *token* [CLS] yang melambangkan mulainya suatu kalimat dan *token* [SEP] yang merupakan *token* untuk memisahkan antar kalimat. Selain itu, juga terdapat *token* [PAD] untuk melakukan penambahan *padding* atau *token* kosong ke dalam kalimat untuk memenuhi *fixed length* dari suatu kalimat yang diinputkan. *Fixed length* pada model BERT adalah 512. Untuk melakukan pelatihan terhadap representasi *bidirectional* maka dilakukan *mask* dengan persentase tertentu dari input *token* secara *random* dan dilanjutkan dengan memprediksi *masked token* yang dilatih pada suatu *pre-training task* dengan *Next Sentence Prediction* (NSP). Lalu, untuk *fine-tuning* model BERT akan melakukan inisialisasi parameter yang sudah dilatih sebelumnya (Devlin et al., 2019). Pada Gambar 2.1 menunjukkan bahwa terdapat perbedaan untuk implementasi model BERT dimana untuk penerapan dalam pembuatan ringkasan dilakukan beberapa perubahan dari model asli BERT. Perubahan terletak pada token [CLS] dan [SEP] yang disisipkan pada setiap awal dan akhir kalimat. Kemudian, saat input dokumen melebihi *fixed length* dari BERT maka akan dilakukan pemotongan terhadap kalimat yang berikutnya. Selain itu, pemberian *segment embedding* dilakukan secara selang-seling pada tiap kalimat. Hal tersebut dilakukan supaya model BERT dapat membedakan antar kalimat. *Position embedding* digunakan untuk

memberikan informasi posisi pada token kata (Liu & Lapata, 2019). Mengenai detail struktur lengkap implementasi dari model BERT untuk pembuatan ringkasan dapat dilihat pada Gambar 2.2.



Gambar 2.2 Detail lengkap untuk struktur BERT Summarization

Sumber: Liu, Y. (2019). Fine-tune BERT for Extractive Summarization. *ArXiv*. http://arxiv.org/abs/1903.10318

2.1.3.1 Token Embeddings

Pada *token embeddings* merupakan proses awal bagi suatu input teks dokumen. Proses yang akan dilakukan adalah mengubah input kata menjadi representasi vektor dengan dimensi yang sudah ditetapkan. Pada BERT, tiap kata akan direpresentasikan sebagai vektor *768-dimensional*. Terdapat 2 tahapan yang dilakukan pada *token embeddings*. Tahap pertama, suatu input teks akan dilakukan proses tokenisasi yaitu memecah kalimat menjadi kata yang kemudian juga disisipkan dengan token [CLS] pada tiap awal kalimat dan [SEP] pada tiap akhir kalimat yang sudah ditokenisasi. Tujuan dari adanya token [CLS] adalah mengumpulkan fitur untuk representasi dari kalimat yang mengikutinya dan [SEP] untuk keperluan pemisahan antar kalimat. Tokenisasi dilakukan dengan menggunakan metode *WordPiece tokenization*. *WordPiece tokenization* merupakan suatu metode yang bertujuan untuk memperoleh keseimbangan antara ukuran kata *vocab* dan *out-of-vocab*. Penggunaan dari *WordPiece tokenization* memampukan BERT untuk menyimpan 30552 kata dalam *vocab* nya. Tahap kedua,

akan dilakukan pengubahan *WordPiece Token* menjadi representasi vektor 768-*dimensional* (Devlin et al., 2019).

Pada penerapan dari *WordPiece tokenization* terdapat karakter spesial ## berfungsi sebagai penanda untuk setiap kata yang disegmentasi kecuali untuk *subword* pertama tidak menggunakan penanda ## (Devlin et al., 2019). Bila terdapat kalimat 'saya sedang memakan nasi goreng dengan mentimun' maka akan diubah menjadi ['saya', 'sedang', 'memakan','nasi','goreng','dengan','ment','##imu','##n']. Bila suatu kata tidak dikenali dalam *dictionary* yang mengubah string menjadi id maka akan dilakukan pemecahan kata menjadi sub-kata dengan sub-kata pertama tanpa diikuti penanda ##.

2.1.3.2 Interval Segment Embeddings

Pada *interval segment embeddings* dilakukan untuk membedakan antar kalimat dengan menggunakan representasi vektor segmen *even/odd* yaitu dengan menyisipkan pelabelan secara selang-seling berdasarkan urutan kalimat terhadap *word embeddings*. Sebagai contoh terdapat input kalimat 'Saya suka memakan ikan goreng. Adik suka memakan ayam goreng.' maka kedua kalimat tersebut akan dilakukan tokenisasi. Setelah itu, dilakukan pelabelan terhadap kalimat tersebut dengan menggunakan 0 untuk kalimat urutan *i* genap dan 1 untuk kalimat urutan *i* ganjil. Pelabelan tersebut merupakan representasi vektor dari segmen kalimat (Devlin et al., 2019).

2.1.3.3 Position Embeddings

Position embeddings atau *position encoding* diperlukan sehingga model BERT dapat mengetahui posisi suatu teks dalam dokumen. Bila terdapat input teks seperti 'walaupun saya sedang lelah, saya tetap bekerja'. Dalam hal ini kata 'saya' pada 'saya sedang lelah' akan memiliki representasi vektor yang berbeda dengan kata 'saya' pada 'saya tetap bekerja' karena memiliki posisi yang berbeda. Tujuan dari *position embeddings* adalah untuk memberikan informasi mengenai posisi absolut dari tiap *token* karena pada struktur *transformer* tidak melakukan *encode* dengan sifat *sequential*.

Persamaan yang digunakan untuk membuat *position embeddings* pada *time step* genap:

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{model}}) \quad (2.1)$$

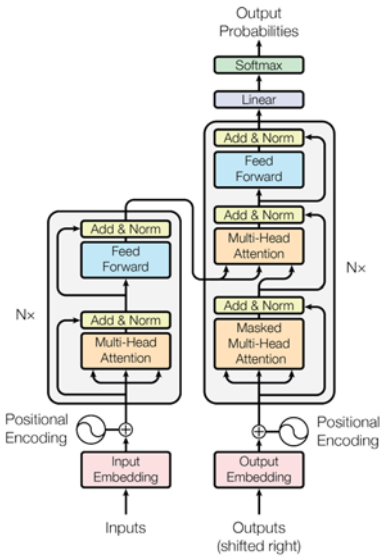
Persamaan yang digunakan untuk membuat *position embeddings* pada *time step* ganjil:

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{model}}) \quad (2.2)$$

Untuk kedua persamaan diatas, *pos* melambangkan posisi dan *i* merupakan dimensi. Tiap dimensi dari *position embeddings* tersusun dari *sinusoidal position*. Model ini dipilih karena akan mempermudah dalam belajar untuk fokus terhadap posisi kalimat (Vaswani et al., 2017).

2.1.4 Transformer

Pada *transformer* terdapat beberapa komponen yaitu *attention* dan *feed forward network*. *Attention* pada *transformer* terdiri atas banyak *head* yang mempunyai *weight* berbeda-beda dan dapat menempati urutan posisi berbeda-beda (Vaswani et al., 2017). Berikut merupakan struktur dari *transformer*.



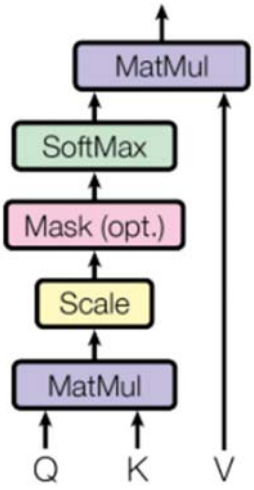
Gambar 2.3 Struktur *transformer*

Sumber: Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017-Decem*(Nips), 5999–6009.

Pada *transformer* terdapat input *embeddings* yang mengubah kata menjadi *list* angka yang merepresentasikan kata yang diinputkan. Lalu, dilakukan *positional encoding* untuk menambahkan informasi mengenai posisi ke *input embeddings* dimana untuk *time step* ganjil akan dibuat dengan fungsi *cos* dan untuk *time step* genap akan dibuat dengan fungsi *sin* seperti penjelasan pada subbab 2.1.3.3. Pada *transformer* tersusun atas *sub-layer* yaitu *multi-head attention* dan *feed forward network*. Lalu, juga terdapat koneksi residual dan layer normalisasi dimana koneksi residual membantu supaya tidak kehilangan informasi dari input sebelumnya sedangkan layer normalisasi untuk menstabilkan nilai ke layer berikutnya sehingga tetap konsisten (Xiong et al., 2020). *Self-attention* dalam model *transformer* menerima *embedding* yang diperoleh dari perkalian antara matriks *query*, *key*, dan *value*. Detail dari proses kalkulasi tersebut dapat dilihat pada persamaan 2.3. Mekanisme *self-attention* digunakan pada model untuk melakukan asosiasi tiap kata dari input ke kata lain (Vaswani et al., 2017).

Self-attention diperoleh dengan menggunakan persamaan *scaled-dot product attention* berikut:

$$Attention\ (Q, K, V) = \ softmax_k\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.3}$$



Gambar 2.4 Diagram *scaled dot product attention*

Sumber: Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, *2017-Decem*(Nips), 5999–6009.

Dimana pada persamaan (*2.3*) akan dilakukan penghitungan skor *attention* menggunakan *Query* (Q), *Key* (K), dan *Value* (V). Tahap awal, akan dilakukan perkalian titik dari matriks *query* dengan matriks *key* dari masing-masing kata. Kemudian, diulangi dengan perkalian titik dari matriks *query* dengan matriks *key* dari kata berikutnya dan seterusnya. Dengan melakukan perkalian akan menghasilkan matriks skor. Skor digunakan untuk mengukur seberapa banyak fokus untuk kata-kata dari urutan input dengan kata pada posisi tertentu. Jadi setiap kata akan memiliki skor yang sesuai dengan kata lain dalam langkah waktu. Semakin tinggi skor, maka akan semakin fokus. Kemudian, akan dilakukan penskalaaan nilai untuk mendapatkan gradien yang lebih stabil dimana akan dilakukan pembagian matriks skor dengan akar kuadrat dari dimensi *key* (d_k). Penskalaan dilakukan sehingga nilai dari hasil perkalian *dot* antara matriks *query* dan *key* tidak menjadi terlalu besar. Matriks berskala tersebut kemudian dilewatkan melalui *softmax* untuk mendapatkan bobot perhatian. Hasil dari *softmax* akan menghasilkan nilai diantara 0 dan 1. Dengan melakukan *softmax*, skor yang lebih tinggi semakin diperkuat dan skor yang lebih rendah akan diperkecil sehingga memungkinkan model untuk lebih memperhatikan tentang kata-kata mana yang akan dipilih. Setelah memperoleh keluaran *softmax*, matriks ini dikalikan dengan matriks *value*. Skor *softmax* yang lebih tinggi akan membuat nilai kata yang dipelajari model menjadi lebih penting sedangkan skor yang rendah akan menunjukkan kata-kata yang tidak relevan (Vaswani et al., 2017). Dalam melakukan klasifikasi terhadap informasi yang diambil akan digunakan *linear layer* untuk *feed forward neural network* dan fungsi *aktivasi sigmoid* (Deng & Liu, 2018).

2.1.5 Recall-Oriented Understudy for Gisting Evaluation (ROUGE)

ROUGE merupakan salah satu cara untuk melakukan evaluasi kualitas dari suatu ringkasan yang dibuat secara otomatis. Ringkasan akan dibandingkan dengan *gold summary* yang ada. *Gold summary* merupakan ringkasan dibuat oleh manusia dan menjadi standar bagi teks yang akan diringkas. Pemilihan *ROUGE* dalam evaluasi ringkasan dikarenakan sudah menjadi standar evaluasi dan telah banyak digunakan oleh penelitian lain. Terdapat 2 jenis *ROUGE* yang digunakan dalam melakukan peringkasan yaitu *ROUGE-N* dan *ROUGE-L*. *ROUGE-N* merupakan pencarian nilai *recall* berdasarkan *n-gram* sedangkan *ROUGE-L* menggunakan *longest common subsequence* (Lin, 2004). *ROUGE-N* digunakan untuk menilai dari sisi *informativeness* atau seberapa banyak kesamaan informasi yang terkandung dalam hasil ringkasan sistem dengan referensi ringkasan. *ROUGE-L* akan menilai dari sisi *fluency* atau

seberapa lancar hasil ringkasan yang dihasilkan dengan melihat urutan kesamaan kata dalam suatu kalimat terhadap referensi ringkasan (Lin, 2004).

Persamaan yang digunakan untuk menghitung *ROUGE-N*:

$$\frac{\sum_{S\epsilon\{ReferenceSummaries\}}\sum_{gram_n\,\epsilon\,S} Count_{match}(gram_n)}{\sum_{S\epsilon\{ReferenceSummaries\}}\sum_{gram_n\,\epsilon\,S} Count(gram_n)} \quad (2.4)$$

Dimana *n* merupakan panjang dari *n-gram* dan $count_{match}(gram_n)$ merupakan jumlah *n-gram* yang cocok pada ringkasan yang dibuat otomatis dengan yang dijadikan referensi. Lalu, $count(gram_n)$ merupakan jumlah total dari *n-gram* yang terjadi dalam referensi ringkasan.

Persamaan yang digunakan untuk menghitung *ROUGE-L*:

$$R_{lcs} = \frac{LCS\ (X,Y)}{m} \quad (2.5)$$

$$P_{lcs} = \frac{LCS\ (X,Y)}{n} \quad (2.6)$$

$$F_{lcs} = \frac{(1+\ \beta^2)R_{lcs}P_{lcs}}{R_{lcs} +\ \beta^2\ P_{lcs}} \quad (2.7)$$

Dimana untuk perhitungan *ROUGE-L* terdapat pada F_{lcs}. Untuk R_{lcs} dan P_{lcs} didapatkan dengan menghitung kesamaan antara kalimat ringkasan referensi X dan kalimat ringkasan sistem Y dimana untuk m merupakan panjang kalimat ringkasan referensi X dan n merupakan panjang kalimat ringkasan sistem Y. Lalu untuk nilai β didapatkan dari P_{lcs}/R_{lcs}. Lalu, untuk nilai *ROUGE-L* didapat lewat perhitungan F_{lcs}.

2.2 Tinjauan Studi

Subbab mengenai penelitian-penelitian yang serupa yang sudah pernah dilakukan sebelum penelitian ini, hal hal yang diangkat seperti masalah yang diangkat dalam penelitian sebelumnya, metode yang digunakan dalam penelitian sebelumnya, hasil yang diperoleh dari penelitian sebelumnya, dan perbedaan penelitian sebelumnya dengan penelitian yang akan dilakukan.

2.2.1 Neural Summarization by Extracting Sentences and Words (Cheng & Lapata, 2016)

Masalah yang diangkat oleh peneliti adalah besarnya jumlah data teks yang ada sehingga terdapat kebutuhan dengan tujuan untuk mempermudah pembacaan dengan mengembangkan peringkasan otomatis dalam membuat ringkasan yang lebih pendek dari dokumen dengan tetap menjaga makna informasi dari dokumen yang diringkas.

Metode yang digunakan pada penelitian ini adalah *neural network* berbasis *encoder* dan *attention-based content extractor* dimana *encoder* berperan untuk mendapatkan makna dari dokumen berdasarkan dari kalimat dan kata pada dokumen tersebut. Kemudian, *attention* akan memilih kalimat atau kata untuk menghasilkan ringkasan ekstraktif.

Hasil dari penelitian ini didapatkan bahwa penerapan dari metode yang diusulkan dalam menghasilkan ringkasan ekstraktif dengan pengukuran *ROUGE* adalah 47.4% untuk *ROUGE-1*, 23% untuk *ROUGE-2*, 43.5% untuk *ROUGE-L* pada dataset DUC 2002 serta 21.2% untuk *ROUGE-1*, 8.3% untuk *ROUGE-2*, dan 12% untuk *ROUGE-L* pada dataset DailyMail.

Perbedaan dari penelitian yang dilakukan oleh peneliti dengan penelitian yang akan dilakukan adalah pada penelitian ini menggunakan dataset berita berbahasa Inggris sedangkan pada penelitian yang akan dilakukan melakukan peringkasan otomatis pada berita berbahasa Indonesia dengan menggunakan dataset penelitian yaitu Indosum.

2.2.2 Peringkasan Ekstraktif Teks Bahasa Indonesia dengan Pendekatan Unsupervised Menggunakan Metode Clustering (Ismi & Ardianto, 2019)

Masalah yang diangkat oleh peneliti adalah volume informasi yang sangat banyak karena perkembangan teknologi informasi yang pesat sehingga harus memilah informasi yang penting untuk merangkum informasi tersebut. Proses meringkas informasi yang banyak juga memerlukan waktu yang lama untuk menyusuri setiap informasi dan mengambil informasi utama.

Metode yang diusulkan oleh peneliti adalah menggunakan metode *K-Means clustering* dalam membuat ringkasan ekstraktif secara otomatis. Sebelumnya akan dilakukan *text preprocessing* dengan *tokenization* (pemisahan dokumen teks untuk mendapatkan frasa dengan inti yang berbobot), *stopword removal* (menghilangkan kata-kata yang tidak memiliki makna), *stemming* (mengidentifikasi bentuk akar kata dengan menghapus imbuhan). Lalu, dilanjutkan dengan pembobotan TF-IDF (bobot suatu istilah akan semakin besar bila sering muncul pada suatu dokumen dan semakin kecil bila muncul dalam banyak dokumen). Kemudian, diterapkan

K-Means clustering dimana setiap klaster dapat terdiri dari beberapa kalimat dan tiap klaster akan mewakili satu kalimat. Satu kalimat tersebut nantinya akan menyusun ringkasan teks secara keseluruhan untuk menghasilkan ringkasan ekstraktif.

Hasil dari penelitian menunjukkan bahwa dengan penerapan *K-Means Clustering* dalam sistem peringkasan ekstraktif otomatis diperoleh nilai *F1-Score* dari *ROUGE-1* sebesar 49.37%, *ROUGE-2* sebesar 38.18% dan *ROUGE-L* sebesar 46.87%.

Perbedaan dari penelitian yang dilakukan oleh peneliti dengan penelitian yang akan dilakukan adalah pada penelitian ini menggunakan metode *clustering* dalam membuat ringkasan berita berbahasa Indonesia sedangkan pada penelitian yang akan dilakukan menggunakan metode BERT dalam menghasilkan ringkasan berita berbahasa Indonesia secara otomatis.

2.2.3 Penerapan Recurrent Neural Network untuk Pembuatan Ringkasan Ekstraktif Otomatis pada Berita Berbahasa Indonesia (Halim et al., 2020)

Masalah yang diangkat oleh peneliti adalah perkembangan internet yang sangat pesat sehingga membuat kewalahan dalam mengikuti informasi yang tersedia secara online. Ringkasan dapat dibuat manual oleh manusia akan tetapi memerlukan waktu yang cukup lama karena informasi yang tersedia sangat banyak. Berita berbahasa Indonesia dipilih karena belum terlalu berkembang bila dibandingkan dengan berita berbahasa Inggris. Kemudian, untuk tingkat kualitas berita berbahasa Indonesia masih cukup rendah dimana kebanyakan masih dibawah 50%.

Metode yang diusulkan oleh peneliti adalah menggunakan metode *Recurrent Neural Network* untuk membuat ringkasan berita berbahasa Indonesia secara otomatis dengan menggunakan dataset Indosum.

Hasil pada penelitian ini adalah diperoleh *F1-Score* sebesar 57.01 untuk *ROUGE-1*, 51.17 untuk *ROUGE-2* dan 55.10 untuk *ROUGE-L* dengan referensi abstraktif serta *F1-Score* sebesar 84.10 untuk *ROUGE-1*, 83.10 untuk *ROUGE-2*, dan 83.31 untuk *ROUGE-L* dengan referensi ekstraktif.

Perbedaan dari penelitian yang dilakukan oleh peneliti dengan penelitian yang akan dilakukan adalah pada penelitian ini menggunakan *Recurrent Neural Network* sedangkan pada penelitian yang akan dilakukan menggunakan metode BERT.

2.2.4 Arabic Text Summarization Using AraBERT Model Using Extractive Text Summarization Approach (Nada et al., 2020)

Masalah yang diangkat oleh peneliti adalah informasi teks yang tersedia secara *online* semakin berkembang dengan pesat sehingga jumlah informasi semakin banyak dan mengakibatkan pembaca kesulitan dalam membaca kumpulan teks tersebut. Selain itu, sebagian besar penelitian masih dilakukan pada teks berbahasa Inggris. Peneliti menyebutkan bahwa pada penelitian teks berbahasa Arab masih rendah. Beberapa permasalahan lainnya adalah kompleksitas dari bahasa Arab dan *library* NLP untuk bahasa Arab masih kurang mendukung.

Metode yang diusulkan oleh peneliti adalah menggunakan metode AraBERT dimana pada input awalnya dilakukan operasi *preprocessing* untuk menghilangkan kalimat yang tidak memiliki makna. Kemudian, dilakukan pengubahan teks menjadi *sentence embedding* sebelum diteruskan pada model AraBERT. Setelah itu, akan dilakukan proses *clustering* dengan menggunakan *K-Means* pada hasil matriks *embedding* dari model AraBERT. Lalu, dari hasil *clustering* akan dipilih kalimat terdekat dengan tiap *cluster centroid* sebagai hasil ringkasan.

Hasil dari penelitian yang telah dilakukan dengan metode yang diusulkan yaitu diperoleh *F-measure* sebesar 0.54 untuk ROUGE-1 dan *F-measure* sebesar 0.51 untuk ROUGE-2 dengan menggunakan pendekatan ekstraktif.

Perbedaan dari penelitian yang dilakukan oleh peneliti dengan penelitian yang akan dilakukan adalah pada penelitian ini dilakukan pada teks berita berbahasa Arab sedangkan pada penelitian yang akan dilakukan menggunakan model BERT pada teks berita berbahasa Indonesia.