Penerapan *Recurrent Neural Network* untuk Pembuatan Ringkasan Ekstraktif Otomatis pada Berita Berbahasa Indonesia

Kristian Halim [1], Henry Novianus Palit [2], Alvin Nathaniel Tjondrowiguno [3]
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121-131. Surabaya 60236
Telp (031) – 2983455, Fax. (031) 8417658
E-mail: halimchristian.halim@gmail.com [1], hnpalit@petra.ac.id [2], alvin.nathaniel@petra.ac.id [3]

ABSTRAK

Saat ini perkembangan informasi sangatlah pesat, dan berita *online* menjadi salah satu dari sumber informasi tersebut. Oleh karena perkembangan yang pesat ini, jumlah berita yang tersedia semakin banyak untuk bisa dibaca oleh manusia sehingga dikembangkanlah sebuah program untuk melakukan pembuatan ringkasan berita agar mengurangi waktu baca dengan memanfaatkan *neural network* sebagai basis program.

Metode yang digunakan adalah *Recurrent Neural Network* untuk melakukan training pada model. Jenis *Recurrent Neural Network* yang digunakan adalah *Gated Recurrent Unit* yang dijalankan 2 kali yaitu pada *word level* dan pada *sentence level*. Dalam pembuatan model *GRU-RNN* ini, dilakukan beberapa percobaan seperti mengganti *weight* awal *word embedding*, pergantian *pooling*, penghilangan *dropout layer*, dan dilakukan beberapa *preprocessing* pada dataset.

Hasil penelitian menunjukan bahwa dengan model awal, hasil *F1 – Score* dari *ROUGE – 1, ROUGE – 2* dan *ROUGE – L* bisa mencapai sekitar 80% apabila dibandingkan dengan referensi ekstraktif dan sekitar 50% apabila dibandingkan dengan referensi abstraktif. Hasil pengujian menunjukan bahwa peforma model paling bagus dengan menggunakan *weight* awal *word embedding* dari dataset *training* dengan menggunakan *average pooling* tanpa diberi *dropout layer*. Hasil terbaik pengujian memberikan *F1 – Score* 84.10 untuk *ROUGE – 1*, 83.10 untuk *ROUGE – 2* dan 83.31 untuk *ROUGE – L* pada referensi ekstraktif dan *F1 – Score* 57.01 untuk *ROUGE – 1*, 51.17 untuk *ROUGE – 2* dan 55.10 untuk *ROUGE – L* pada referensi abstraktif.

Kata Kunci: *Natural Language Processing, Recurrent Neural Network*

ABSTRACT

Technology advancement in modern world allow huge amount of information to flow everyday and news became one of the source to get that information. Because of this advancement, available information through news have been increased and so program is develop to make summary of news to reduce reading time using the neural network as the basis of this program.

The method used for training the model is Recurrent Neural Network. The type of Recurrent Neural Network that being used is Gated Recurrent Unit that is run in 2 level, the word level and then the sentence level. As for making the Recurrent Neural Network model, some experiment can be carried out, like changing initial weight of the word embedding, change the pooling method, removing dropout layer, and some preprocessing for the dataset.

The results shows that for the initial model, F1 – Score for ROUGE – 1, ROUGE – 2, and ROUGE – L can reach up to 80% when using extractive summary as the reference and up to 50% when using abstractive summary as the reference. The experiment shows that the best model is using training dataset as the initial word embedding weight, using average pooling and removing the dropout layer. The best experiment result gives F1 – Score 84.10 for ROUGE – 1, 83.10 for ROUGE – 2 and 83.31 for ROUGE – L using the extractive reference and 57.01 for ROUGE – 1, 51.17 for ROUGE – 2 and 55.10 for ROUGE – L using the abstrative reference.

Keywords: *Natural Language Processing, Recurrent Neural Network*

1. PENDAHULUAN

Informasi adalah salah satu aspek yang penting dalam kehidupan sehari - hari manusia dan salah satu cara untuk mendapatkan informasi adalah dengan melakukan kegiatan membaca. Informasi bisa didapatkan melalui berbagai cara, salah satunya adalah *web*. Namun karena perkembangan internet yang sangat pesat, manusia kewalahan untuk bisa terus mengikuti berbagai informasi yang tersedia secara online [1]. Untuk mengatasi hal ini, maka diperlukan pembuatan ringkasan dari informasi – informasi tadi. Ringkasan sendiri adalah suatu teks yang diproduksi dari kumpulan teks lainnya yang masih memiliki informasi penting dari teks *original* dan juga panjangnya jauh lebih pendek daripada teks *original* [7].

Dari berbagai macam sumber untuk bisa mendapatkan informasi di internet, salah satu sumber yang mengandung banyak sekali informasi adalah berita *online*. Berita mengandung informasi – informasi yang memiliki beraneka ragam topik, mulai dari teknologi, politik, hingga olahraga. Karena variasi informasi yang luas dan cukup sering digunakan, berita menjadi objek yang sesuai untuk penelitian ini. Berita berbahasa Indonesia dipilih karena pembuatan ringkasan otomatis untuk berita berbahasa Indonesia belum terlalu berkembang dibandingkan dengan berita berbahasa Inggris.

Berdasarkan hasilnya, metode pembuatan ringkasan otomatis dibedakan menjadi 2 kategori, yaitu secara ekstraktif dan abstraktif. Metode ekstraktif memiliki tujuan utama untuk membuat ringkasan dengan cara memilih kata – kata dan kalimat – kalimat yang penting dari suatu dokumen, sementara metode abstraktif memiliki tujuan utama untuk membuat parafrase dari informasi yang didapat pada suatu dokumen [6]. Metode yang dilakukan pada penelitian ini yaitu *Recurrent Neural Network* termasuk dalam kategori metode ekstraktif. Alasan dari penggunaan metode yang ekstraktif ini adalah karena para

pengguna internet cenderung lebih memilih untuk membaca teks ringkasan yang memiliki bentuk yang mirip dengan teks original yang dibuat oleh penulisnya [2]. Selain itu metode – metode lain yang digunakan untuk membuat ringkasan berita Indonesia secara ekstraktif seperti pada [4] yang menggunakan algoritma *TextTeaser* dan menghasilkan kesimpulan bahwa metode buatan mereka masih perlu perbaikan.

2. TINJAUAN STUDI

Ada banyak cara untuk membuat ringkasan secara ekstraktif. Salah satu pendekatan yang bisa dilakukan adalah dengan menggunakan *Recurrent Neural Network* (*RNN*). Untuk penelitian ini, digunakan model yang di kemukakan pada [6] dimana *RNN* mempunyai 2 layer dan sifatnya *bi-directional*. Seperti yang tertera pada gambar, tiap sentence awalnya akan dipecah menjadi tiap *word* dengan menggunakan *word embedding*. *Word embedding* adalah suatu teknik pemetaan kata – kata berdasarkan suatu kamus yang sudah ada. Hasil pemetaan ini berupa vektor – vektor yang berisi angka *real*. Hasil dari *word embedding* ini kemudian akan dihitung dalam *hidden state* dari layer pertama dari *RNN* yaitu *word-level*. Pada gambar terlihat ada panah bolak-balik pada layer yang sama. Ini menandakan *bi-directional* yang artinya ada *RNN* satu lagi pada layer tersebut untuk melakukan perhitungan *hidden state* namun dari belakang ke depan (kebalikan). Setelah kedua *RNN* di *word-level* selesai, hidden layer pada *RNN* di *word-level* tersebut di rata-rata dan ditambah untuk dimasukan ke dalam *hidden state* pada *RNN* di *sentence-level* untuk kemudian dihitung dengan cara *bi-directional.* Setelah perhitungan pada tiap *hidden state* milik *RNN* di *sentence-level* selesai, berikutnya adalah menentukan apakah kalimat tersebut cocok untuk dimasukkan summary atau tidak dengan menggunakan *sigmoid activation based classification layer* yang akan memberi keputusan apakah suatu kalimat akan dimasukan ke dalam ringkasan atau tidak. Keputusan yang diambil akan di tentukan oleh logistic layer yang mempertimbangkan 5 hal yaitu konten kalimat (*content*), ciri khas kalimat (*salience*), ke-orisinalan kalimat (*novelty*), posisi absolut kalimat pada dokumen (*absolute position*) dan posisi relative kalimat pada dokumen (*relative position*) [6].

Dalam *RNN* biasa, terdapat permasalahan yang disebut dengan *vanishing gradient problem*, dimana pada suatu saat perubahan *gradient* dalam suatu model *RNN* menjadi sangat kecil sehingga *weight* tidak mengalami perbuahan. Untuk mengatasi permasahalan ini, *GRU* menggunakan *gate* yang disebut dengan *update gate* dan *reset gate*. *Update gate* berfungsi untuk menentukan seberapa banyak informasi dari iterasi – iterasi sebelumnya yang ingin dipertahankan untuk digunakan di masa depan, sementara *Reset Gate* berfungsi untuk menentukan seberapa banyak informasi dari iterasi – iterasi sebelumnya yang ingin dilupakan [3].

ROUGE adalah salah satu cara untuk mengevaluasi kualitas dari suatu ringkasan yang dibuat secara otomatis dengan membandingkannya dengan *gold summary* yang ada. *Gold summary* adalah suatu ringkasan yang dibuat oleh manusia dan menjadi standar ringkasan bagi teks yang ingin diringkas tersebut. Alasan *ROUGE* dipilih sebagai penentu kualitas adalah karena *ROUGE* sudah menjadi metode *state-of-the-art* dalam melakukan evaluasi kualitas dari suatu ringkasan dan banyak sekali digunakan oleh penelitian – penelitian lain sehingga dapat ROUGE dapat digunakan dengan untuk melakukan perbandingan metode.

3. DESAIN SISTEM

3.1 Dataset and Preprocessing

Dataset berita berbahasa Indonesia diambil dari dataset yang telah disediakan pada [5]. Dataset dalam format JSON dan dipecah menjadi 3 bagian, yaitu *Train*, *Test*, dan *Dev*. Selain dipecah menjadi 3 bagian, dataset yang disediakan juga dibagi menjadi 5 fold, dengan masing – masing memiliki data sebanyak 18.774 ribu berita. Dataset *Train* digunakan untuk melakukan proses *training*, *Test* untuk proses *test*, dan *Dev* untuk proses evaluasi saat *training*.

Sebelum digunakan dalam program, data perlu diolah terlebih dahulu agar mudah untuk diintegrasikan ke dalam program. Berikut adalah beberapa hal yang dilakukan dalam pengolahan:

1. Mengambil bagian ‘gold_labels’, ‘paragraphs’, dan ‘summary’ dari data original. 3 bagian ini diambil karena alasan berikut: ‘gold_labels’ diambil sebagai dasar dalam melakukan *training* dan juga ketika melakukan pengujian dengan referensi ekstraktif, ‘paragraphs’ berisi berita yang akan diringkas sehingga tentunya perlu diambil dan ‘summary’ berisi ringkasan abstraktif yang dibuat oleh manusia dan digunakan saat melakukan pengujian dengan referensi abstraktif.
2. Membuka array yang ada pada bagian ‘summary’ untuk disusun menjadi satu kesatuan document berita. Tiap kalimatnya dipisahkan dengan delimiter ‘\z’.
3. Membuka array yang ada pada bagian ‘gold_labels’ untuk disusun menjadi string yang antara elementnya dipisahkan dengan delimiter ‘\n’.

3.2 Word Embedding

Untuk pembuatan *word embedding* pada kata – kata berbahasa Indonesia, diperlukan terlebih sumber yang menyediakan banyak kata – kata berbahasa Indonesia. Untuk sumber pembuatan *word embedding* pada penelitian ini dipilih dari *Wikipedia* berbahasa Indonesia dan juga dari dataset train. Sementara untuk dataset train diambil untuk semua 5 *fold*. Setelah mendapatkan dump dari *Wikipedia* dan dari dataset train, perlu dilakukan pengekstrakan teks artikel yang ada di dalam dump. Lalu untuk memudah pembacaan, semua teks artikel yang didapat di save ke dalam text file dengan ‘\n’ atau baris baru sebagai pemisah dari tiap artikel. Setelah itu, dibaca lagi baris demi baris untuk dicari word embeddingnya. Apabila *word embedding* dari tiap artikel sudah terbuat, maka model *word embedding* bisa di save.

Word embedding yang di bentuk dari artikel *Wikipedia* maupun dataset train ini nantinya digunakan sebagai *weight* awal dari *word embedding* yang akan digunakan dalam *training* pada *neural network*.

3.3 Training dan Model RNN

Proses *training* dimulai dengan menerima inputan berupa model *word embedding*, *word2id*, dan dataset untuk *training* dan *evaluation*. Setelah itu perlu dilakukan inisialisasi model *RNN*, *optimizer*, dan *loss function*. Untuk *optimizer* digunakan *Adam optimizer*, dan untuk *loss function* digunakan *Binary Cross Entropy*. Setelah semua proses load dan inisialisasi selesai, mulai dilakukan training yang akan berlangsung selama beberapa iterasi / *epoch*. Untuk setting awal, jumlah *epoch* di set 5 dengan alasan karena dengan jumlah tersebut bila dipakaikan ke data berita berbahasa Inggris, nilainya sudah bagus. Dalam satu iterasi *epoch*

proses di dalamnya di jalankan beberapa kali tergantung jumlah *batch* yang ada.

Training dimulai dengan mencari *feature*, label ringkasan, dan jumlah kalimat untuk tiap data pada *batch* tersebut. Kemudian training dilanjutkan dengan mem forward-feedkan model *RNN*. Setelah itu, akan dilakukan pencarian *loss* menggunakan *loss function* dengan label target sebagai ukuran kebenaran. *Loss* yang didapat kemudian digunakan untuk update *weight*.

Model terdiri dari 2 *layer Bi-Directional GRU-RNN* dimana *layer* pertama dijalankan pada *level* kata dan *layer* kedua dijalankan pada *level* kalimat yang dilanjutkan ke dalam *layer* klasifikasi. Tiap *layer RNN* tadi didalamnya juga terdapat *dropout layer*. Hasil dari tiap *layer* akan dilakukan *average pooling* terlebih dahulu sebelum dilanjutkan untuk masuk ke *layer* berikutnya. *Layer* klasifikasi dihitung bedasarkan 5 parameter yaitu:

- *Content* → berkaitan dengan kualitas konten dari kalimat yang bersangkutan.
- *Salience* → berkaitan dengan seberapa 'penting' kalimat yang bersangkutan bedasarkan artikel berita dari kalimat tersebut.
- *Novelty* → berkaitan dengan seberapa 'original' / redundant kalimat yang bersangkutan bedasarkan artikel berita dari kalimat tersebut.
- *Absolute position* → berkaitan dengan posisi suatu kalimat dari keseluruhan artikel.
- *Relative position* → berkaitan dengan posisi suatu kalimat dari segment yang ada pada artikel. Segment merupakan representasi jumlah tertentu yang sudah ditentukan secara fix (untuk setting awal ditentukan 10) yang memisahkan artikel menjadi beberapa segment tertentu.

Setelah mendapatkan nilai dari tiap *layer* klasifikasi, semuanya akan dihitung dengan menggunakan fungsi sigmoid yang akhirnya akan menghasilkan suatu array yang mana array ini menyimpan angka double antara 0 hingga 1 yang menunjukan probabilitas dari tiap kalimat untuk bisa masuk ke dalam ringkasan / *summary*.

Berikut adalah penjelasan cara pencarian dan penggunaan dari masing – masing komponen yang diperlukan dalam training:

- *Feature* → *Feature* digunakan sebagai input dari *RNN*. *Feature* yang dimaksud disini adalah id unik dari tiap kata yang ada pada artikel berita.
- Label Ringkasan → Label Ringkasan merupakan kumpulan angka antara 0 atau 1 yang menunjukan apakah suatu kalimat pada artikel terdapat pada ringkasan yang dibuat manusia atau tidak.
- Jumlah Kalimat → Jumlah kalimat ini digunakan untuk penentuan posisi *absolute* dan *relative* pada *layer* klasifikasi di dalam model *RNN*.

Untuk proses evaluasi alurnya mirip dengan training hanya saja model tidak di train tapi langsung digunakan untuk mencari *loss* dari dataset evaluasi. Nantinya semua *loss* dari dataset evaluasi akan di rata-rata kemudian *loss* yang sudah di rata-rata tadi digunakan untuk menentukan apakah model semakin baik atau tidak. Tujuan proses evaluasi ini dilakukan adalah dengan harapan agar model tidak mengalami *overfitting* terhadapa dataset training.

3.4 Test

Proses test secara garis besar mirip dengan proses training. Untuk langkah awal, diperlukan mengload model *word embedding*, *word2id*, dataset test dan juga hasil save dari model *RNN* yang sebelumnya sudah di training. Setelah itu dataset test akan dipisah menjadi beberapa *batch* dan untuk tiap batch akan dicari *feature*, label ringkasan, dan jumlah kalimat untuk tiap artikel pada batch bersangkutan dan kemudian dimasukan ke dalam model yang sudah di load di awal untuk menghasilkan *array* yang isinya probabilitas dari tiap kalimat pada suatu artikel. Kemudian tiap artikel / data yang sudah didapatkan probabilitasnya akan diambil probabilitas terbagus sejumlah 'n' untuk dijadikan ringkasan.

4. PENGUJIAN SISTEM

4.1 Konfigurasi Pengujian

Pengujian akan dilakukan sebanyak 2 kali, yang pertama dengan menggunakan ringkasan abstraktif dari dataset (tertera 'summary') sebagai referensi dan yang kedua dengan menggunakan kalimat yang diambil langsung dari berita yang disesuaikan dengan *gold label* yang ada di dataset sebagai referensi. Dataset yang disediakan oleh Kurniawan dan Louvan ini dibuat menjadi 5 *fold*, namun untuk memperpendek waktu beberapa pengujian hanya akan dibandingkan menggunakan *fold* yang pertama saja.

Untuk semua pengujian, jumlah kalimat yang dijadikan sebagai ringkasan adalah 3 kalimat dengan probabilitas tertinggi untuk menjadi ringkasan. Alasan hanya menggunakan 3 kalimat adalah karena pada pengujian yang dilakukan pada [5] dalam mengetes metode – metode untuk dataset mereka, mereka menggunakan 3 kalimat untuk setiap metodenya, sehingga untuk menyamakan penelitian ini juga menggunakan 3 kalimat untuk pengujian. Ketiga kalimat yang dipilih akan disusun sesuai urutan mereka sebenarnya pada berita untuk dijadikan ringkasan.

Untuk pengujian di penelitian ini, penilaian *ROUGE* menggunakan 95 % *confidence interval* mengikuti penilaian *ROUGE* yang digunakan di penelitian - penelitian lain yang serupa, termasuk yang digunakan pada penelitian yang dilakukan Kurniawan dan Louvan dalam pembuatan dataset. Nilai *ROUGE* yang diambil adalah *F1 – Score* dari *ROUGE – 1, ROUGE – 2,* dan *ROUGE – L*.

4.2 Pengujian Awal

Untuk pengujian awal, model yang digunakan mengikuti struktur yang sama dengan yang tertera pada bab 3.

Tabel 1. F1 - Score Pengujian Awal dengan Menggunakan Referensi Abstraktif

Fold	ROUGE – 1	ROUGE – 2	ROUGE – L
1	56.00	50.17	54.11
2	57.13	51.24	55.18
3	56.87	50.94	54.98
4	57.28	51.46	55.40
5	56.63	50.90	54.95
Average	56.91	50.94	54.92

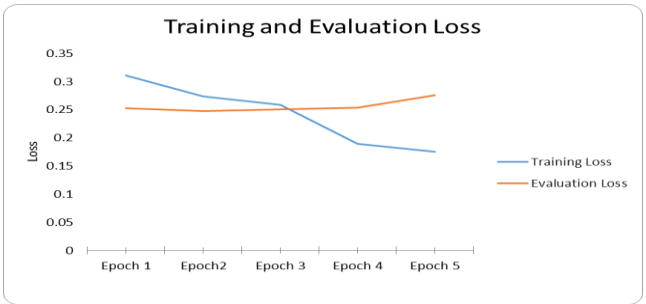
Tabel 2. F1 - Score Pengujian Awal dengan Menggunakan Referensi Ekstraktif

Fold	ROUGE – 1	ROUGE – 2	ROUGE – L
1	82.41	81.05	81.44
2	84.59	83.41	83.51
3	84.03	82.73	83.11
4	84.33	83.16	83.37
5	84.13	82.94	83.29
Average	83.90	82.66	82.94

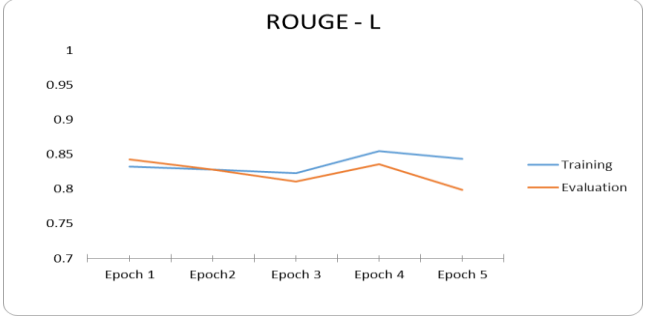
Hasil *F1 – Score* dari *ROUGE* pada pengujian awal dapat dilihat pada tabel 1 dan tabel 2. Dari hasil *F1 – Score* yang didapat dapat dilihat bahwa perbedaan nilai *ROUGE* antara yang menggunakan abstraktif sebagai referensi dengan yang menggunakan ekstraktif sebagai referensi mempunyai perbedaan yang cukup signifikan dengan nilai *ROUGE* ekstraktif hampir 30% diatas abstraktif. Ini dikarenakan refensi abstraktif tidak sepenuhnya mengambil kalimat dari berita yang asli dimana beberapa kata pertama dihilangkan. Namun untuk penelitian ini tetap digunakan keduanya sebagai referensi karena selain untuk mencari tahu peforma ringkasan dari model terhadap ringkasan ekstraktif yang diambil dari berita, peforma ringkasan dari model terhadap ringkasan abstraktif yang ditulis manusia juga ingin dicari tahu.

Selain itu dapat dilihat juga bahwa untuk *fold* 1 hingga 5, perbedaannya tidak terlalu signifikan, sehingga selain keterbatasan waktu, perbedaan score yang tidak signifikan ini juga menjadi alasan untuk pengujian yang lain hanya membandingkan *fold* saja. Untuk peforma dari model dapat dilihat pada gambar 2 bahwa hasil *F1 – Score* dari *ROUGE* memiliki hasil yang cukup tinggi, bahkan ketika menggunakan ringkasan abstraktif sebagai referensi, yaitu sekitar 50% untuk tiap *ROUGE* nya.

Untuk peforma dari model dapat dilihat bahwa hasil *F1 – Score* dari *ROUGE* memiliki hasil yang cukup tinggi, bahkan ketika menggunakan ringkasan abstraktif sebagai referensi, yaitu sekitar 50% untuk tiap *ROUGE* nya.



Gambar 1. Grafik loss training dan evaluation



Gambar 2. F1 – Score ROUGE – L pada data training dan evaluation.

Dari gambar 1 mengenai grafik *loss* dapat dilihat bahwa seiring bertambahnya *epoch*, *loss* dari *training* terus menurun sementara *loss* dari *evaluation* awalnya menurun kemudian mulai naik dan terus naik. Hal ini menunjukan terjadinya *overfitting* dimana model belajar terlalu banyak dari dataset *training* sehingga ketika dicoba ke data yang lain hasilnya (dalam hal ini *ROUGE score*) mengalami penurunan. Oleh karena itu dalam proses training, dilakukan proses evaluasi dan model hanya disimpan ketika *loss* dari evaluation dataset dalam keadaan minimum. Sebagian besar *loss* pada pengujian mencapai minimum pada *epoch* 2 atau 3, sehingga untuk tiap pengujian tetap menggunakan batasan *epoch* yang awal yaitu 5. Pada gambar 2 dapat kita lihat hasil *ROUGE* pada data *training*.

4.3 Pengujian Word Embedding

Pada pengujian awal *word embedding* yang digunakan sebagai inisialisasi *weight* dibuat dari data testing sendiri. Untuk pengujian kali ini dilakukan 2 kali percobaan yaitu mengganti *word embedding* dengan data yang diambil dari *Wikipedia* dan tidak diberi *word embedding.*

Tabel 3. F1 - Score untuk Pengujian Word Embedding pada Referensi Abstraktif

Word Embedding	ROUGE – 1	ROUGE – 2	ROUGE – L
Train Dataset	56.00	50.17	54.11
Wikipedia	55.92	50.12	54.02
None	56.53	50.82	54.53

Tabel 4. F1 – Score untuk Pengujian Word Embedding pada Referensi Ekstraktif

Word Embedding	ROUGE – 1	ROUGE – 2	ROUGE – L
Train Dataset	82.41	81.05	81.44
Wikipedia	82.50	81.18	81.63
None	83.22	82.00	82.23

Dari hasil pengujian pada tabel 3 dan tabel 4 dapat dilihat bahwa ternyata apabila *word embedding* model tidak diberi *weight* awal, hasil *F1 – Score* pada dataset test lebih tinggi dibandingkan dengan ketika diberi *weight* awal, sementara untuk yang diberi *weight* awal yaitu dari *Wikipedia* maupun dataset *train* perbedaannya tidak terlalu jauh.

4.4 Pengujian Dropout Layer

Pada pengujian ini dicoba menghilangkan *dropout layer* dengan probabilitas sebanyak 0.6 yang ada di model pada pengujian awal. Alasan dilakukan percobaan ini adalah karena diperkirakan permasalahan *overfitting* sudah lebih teratasi dengan adanya proses evaluasi di dalam training sehingga penghilangan *dropout layer* mempertahankan semua informasi yang ada dan diharapkan bisa memperbaiki nilai *ROUGE*.

Tabel 5. F1 – Score untuk Pengujian Penghilangan Dropout Layer pada Referensi Abstraktif

Word Embedding	ROUGE – 1	ROUGE – 2	ROUGE – L
Train Dataset	56.92	51.14	54.88
Wikipedia	56.45	50.62	54.48
None	55.60	49.75	53.67

Tabel 6. F1 – Score untuk Pengujian Penghilangan Dropout Layer pada Referensi Ekstraktif

Word Embedding	ROUGE – 1	ROUGE – 2	ROUGE – L
Train Dataset	83.75	82.52	82.70
Wikipedia	83.23	81.93	82.28
None	82.42	81.01	81.45

Dari hasil pengujian pada tabel 5 dan tabel 6 dapat disimpulkan bahwa ternyata bila *dropout layer* dihilangkan dari *RNN* pada *word* dan *sentence level*, hasilnya berbanding terbalik dengan ketika diberi *dropout layer*, dimana model yang memiliki *weight* awal menghasil *F1 – Score* yang lebih baik dibandingkan dengan

yang tidak diberi *weight* awal, dengan hasil *F1 – Score* terbaik sejauh ini dipegang oleh *weight* awal dari dataset train tanpa *dropout layer*.

4.5 Pengujian Pooling

Pada pengujian ini dicoba melakukan pergantian dari *average pooling* ke *maximum pooling.* Alasan dilakukan pengujian ini adalah diperkirakan dengan menggunakan *maximum pooling* dapat diambil satu fitur saja yang paling besar, mengurangi informasi yang didapat untuk dilanjutkan ke layer berikutnya daripada harus mengambil semua informasi yang ada lalu di rata – rata.

Tabel 7. F1 - Score untuk Pengujian Menggunakan Maximum Pooling pada Referensi Abstraktif

Word Embedding	ROUGE – 1	ROUGE – 2	ROUGE – L
Train Dataset No Dropout	56.59	50.79	54.62
Train Dataset With Dropout	54.34	48.36	52.59
Wikipedia No Dropout	56.06	50.24	54.13
Wikipedia With Dropout	55.10	49.31	53.34
None No Dropout	55.47	49.65	53.54
None With Dropout	55.82	49.97	53.83

Tabel 8. F1 - Score untuk Pengujian Menggunakan Maximum Pooling pada Referensi Ekstraktif

Word Embedding	ROUGE – 1	ROUGE – 2	ROUGE – L
Train Dataset No Dropout	83.52	82.24	82.52
Train Dataset With Dropout	80.24	78.70	79.36
Wikipedia No Dropout	82.65	81.37	81.73
Wikipedia With Dropout	81.42	80.06	80.63
None No Dropout	81.87	80.48	80.95
None With Dropout	82.10	80.74	81.13

Dari hasil pengujian pada tabel 7 dan tabel 8 ini dapat dilihat sekilas bahwa efek *dropout layer* dari pengujian di subbab 4.4 memiliki efek yang sama dimana yang diberi *weight* awal lebih baik dengan yang tidak diberi *weight awal* bila *dropout layer* dihilangkan. Sementara untuk pergantian dari *average pooling* ke *maximum pooling* secara keseluruhan menurunkan *F1 – Score* dari ringkasan yang dibuat model, meskipun penurunan yang terjadi tidak terlalu signifikan dengan yang diberi *dropout* mengalami penurunan lebih banyak ketika di *maximum pooling* daripada yang tidak diberi *dropout*. Dari hasil ini dapat disimpulkan bahwa ternyata informasi – informasi yang bisa didapat dari hasil RNN semuanya tergolong penting, bukan hanya nilai terbesar saja.

Hasil percobaan ini pula yang menjadi alasan mengapa pada pengujian *dropout* di subbab 4.4, menghilangkan *dropout* meningkatkan peforma dari model karena *dropout* menghilangkan beberapa informasi yang bisa didapat padahal semua informasi tersebut tergolong penting untuk dimasukan ke dalam penghitungan.

4.6 Pengujian Preprocessing

Pada pengujian ini, dilakukan *preprocessing* pada dataset sebelum dimasukkan ke dalam model. Dataset *preprocessing* ini hanya digunakan dalam *forward feed* dari model ketika melakukan *training*, *test*, ataupun *evaluation*. Hasil ringkasan yang dihasilkan tetap ekstraktif dari dataset yang tidak *preprocessing*. *Preprocessing* yang dilakukan adalah *stopword removal*, *stemming* dan juga menghilangkan beberapa kata dari kalimat – kalimat awal pada berita tertentu. Berita tertentu disini adalah berita yang kalimat awalnya mengandung nama kota dan sumber berita Pengujian ini dilakukan dengan model *RNN* yang hasilnya paling bagus dari pengujian – pengujian sebelumnya, yaitu *weight* awal dari dataset *train*, tidak menggunakan *dropout layer*, dan menggunakan *average pooling.* Pengujian ini dilakukan untuk bisa mengetahui seperti apa efek dari *preprocessing* pada peforma model dalam membuat ringkasan.

Tabel 9. F1 - Score dari Preprocessing dengan Referensi Abstraktif

Preprocessing	ROUGE - 1	ROUGE - 2	ROUGE - L
Tanpa preprocessing	56.92	51.14	54.88
Stopword removal	56.77	51.03	54.77
Stemming	56.25	50.46	54.27
Penghilangan kata awal	56.67	50.87	54.56

Tabel 10. F1 - Score dari Preprocessing dengan Referensi Ekstraktif

Preprocessing	ROUGE - 1	ROUGE - 2	ROUGE-L
Tanpa preprocessing	83.75	82.52	82.70
Stopword removal	83.56	82.33	82.54
Stemming	83.05	81.76	82.10
Penghilangan kata	83.50	82.22	82.41

Dari hasil pengujian pada tabel 9 dan tabel 10 ini dapat disimpulkan bahwa ternyata dengan dilakukan *preprocessing*, tidak terlihat adanya perubahan peforma yang signifikan (terlihat peforma menurun namun, jumlah penurun relatif sedikit). Hal ini diduga karena sekalipun dataset telah di *preprocess*, struktur kalimat pada berita tidak berubah banyak sehingga model tidak mengalami perbedaan belajar yang banyak juga ketika mempelajari berita asli dan yang sudah di *preprocess*.

4.7 Diskusi

Secara garis besar, dari berbagai pengujian yang dilakukan dapat disimpulkan beberapa hal berikut:

- Dari pengujian awal hingga pengujian akhir, hasil yang didapat tidak berbeda terlalu jauh antara pengujian yang satu dengan pengujian yang lain, menandakan bahwa model sudah tergolong stabil, hanya saja perlu dilakukan beberapa *tuning* dan *preprocessing* dataset.
- *Weight* awal *word embedding* yang menghasilkan model terbaik sejauh ini adalah *word embedding* yang dibentuk dari dataset *training*, dimana *word embedding* ini secara

konsisten lebih baik dari yang lain kecuali ketika *dropout layer* digunakan dimana tidak diberi *weight* awal lebih baik.

- Pergantian dari *average pooling* menjadi *maximum pooling* tidak meningkatkan peforma model, karena diduga dengan *maximum pooling* informasi yang didapat menjadi lebih sedikit. Hal yang serupa terjadi dengan pembuangan *dropout layer* dimana ketika *dropout layer* dari model awal dibuang hasilnya lebih baik karena informasi yang didapat lebih banyak.
- *Preprocessing* yang dilakukan pada dataset tidak memberikan perbedaan yang signifikan pada model.
- Secara keseluruhan hasil terbaik dihasilkan dengan menggunakan *training* sebagai *weight* awal memakai *average pooling* tanpa *dropout layer* dan tanpa dilakukan preprocessing.

Berikut dilakukan perbandingan nilai *F1 – Score* untuk tiap *ROUGE* antara model pada percobaan ini dengan model – model lain yang tertera pada percobaan yang dilakukan oleh Kurniawan dan Louvan yang mereka gunakan untuk mengevaluasi dataset mereka. Untuk keperluan perbandingan, model dari dari hasil pengujian terbaik diaplikasikan untuk semua 5 *fold.*

Tabel 11. ROUGE – Score dari Metode Lain

		R-1	R-2	R-L
	ORACLE	79.27 (0.25)	72.52 (0.35)	78.82 (0.28)
	LEAD-3	62.86 (0.34)	54.50 (0.41)	62.10 (0.37)
Unsupervised	SUMBASIC [14], [15]	35.96 (0.18)	20.19 (0.31)	33.77 (0.18)
	LSA [16], [17]	41.37 (0.19)	28.43 (0.25)	39.64 (0.19)
	LEXRANK [18]	62.86 (0.35)	54.44 (0.44)	62.10 (0.37)
	TEXTRANK [20]	42.87 (0.29)	29.02 (0.35)	41.01 (0.31)
Non-neural supervised	BAYES [6]	62.70 (0.39)	54.32 (0.46)	61.93 (0.41)
	HMM [21]	17.62 (0.11)	4.70 (0.11)	15.89 (0.11)
	MAXENT [22]	50.94 (0.42)	44.33 (0.50)	50.26 (0.44)
Neural supervised	NEURALSUM [12]	67.60 (1.25)	61.16 (1.53)	66.86 (1.30)
	NEURALSUM 300 emb. size	**67.96 (0.46)**	**61.65 (0.48)**	**67.24 (0.47)**
	NEURALSUM + FASTTEXT	67.78 (0.69)	61.37 (0.93)	67.05 (0.72)

Tabel 12. Hasil Terbaik dari Penelitian Ini dengan Referensi Abstraktif

Fold	ROUGE - 1	ROUGE - 2	ROUGE - L
1	56.92	51.14	54.88
2	57.30	51.39	55.28
3	56.91	50.91	55.03
4	57.27	51.49	55.43
5	56.68	50.95	54.90
Average	**57.01**	**51.17**	**55.10**

Tabel 13. Hasil Terbaik dari Penelitian Ini dengan Referensi Ekstraktif

Fold	ROUGE - 1	ROUGE - 2	ROUGE - L
1	83.75	82.52	82.70
2	84.59	83.43	83.44
3	84.22	82.87	83.28
4	84.66	83.54	83.73
5	83.31	83.14	83.44
Average	**84.10**	**83.10**	**83.31**

Yang digunakan sebagai pembanding adalah hasil ketika menggunakan ringkasan abstraktif sebagai referensi. Dari perbandingan ini dapat dilihat bahwa peforma model pada penelitian ini terletak kurang lebih di bagian tengah diantara model – model lain yang digunakan untuk mengevaluasi dataset ini. Hasil model – model lain dapat dilihat pada tabel 11 sementara hasil penelitian ini pada tabel 12 dan tabel 13. Namun perbandingan ini tidak bisa dipakai langsung karena diduga *environment* dan cara pengujian yang dilakukan antara penelitian ini dengan yang dilakukan pada [5] memiliki perbedaan. Hal ini dapat dibuktikan dari perbandingan antara referensi ekstraktif dan abstraktif sendiri (tanpa model, *preprocessing*, dan lain – lain) memiliki hasil *F1 – Score* sekitar 60%, padahal pada pengujian yang dilakukan Kurniawan dan Louvan, perbandingan ini (tertera *oracle* pada penelitian mereka) bisa mencapai 70%.

Selain itu, sebelumnya ingin dilakukan pengujian dengan meminta penilaian dari manusia terhadap ringkasan yang dihasilkan. Namun pengujian ini tidak jadi dilakukan dengan alasan karena pengujian demikian akan sulit untuk dievaluasi karena penilaian dari manusia bersifat subjektif. Selain itu referensi abstraktif yang sudah tersedia di dataset pun sudah dibuat manusia, sehingga dengan menggunakan referensi ini dinilai sudah cukup untuk mewakili.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil perancangan model dan penelitian ini, dapat diambil beberapa kesimpulan antara lain:

- Model yang dihasilkan dari *training* dipilih sesuai *loss* terkecil dari dataset *evaluation* untuk mengurangi *overfitting*.
- *Weight* awal untuk *word embedding* yang paling baik secara keseluruhan dibuat bedasarkan dataset *testing*, sementara tanpa *weight* awal baik pada kondisi tertentu dan *Wikipedia* tidak pernah mengalahkan keduanya.
- Penggunaan *maximum pooling* memperburuk peforma model dibandingkan dengan menggunakan *average pooling*.
- Menghilangkan *dropout layer* dapat meningkatkan peforma model dibandingkan dengan menggunakan *dropout layer* dengan probabilitas 0.6.
- *Preprocessing* pada dataset sebelum di proses memperjelek peforma model namun dalam jumlah yang sangat kecil sehingga dapat dibilang bahwa *preprocessing* tidak berpengaruh terlalu banyak pada hasil ringkasan.
- Hasil terbaik didapat dengan menggabungkan dataset *train* sebagai *weight* awal *word embedding*, dengan *average pooling*, tanpa *dropout*, dan di tanpa dengan dataset yang tidak di *preprocess*.
- Hasil terbaik yang bisa didapat menghasilkan nilai *F1 – Score* 57.01 untuk *ROUGE – 1*, 51.17 untuk *ROUGE – 2* dan 55.10 untuk *ROUGE – L* dengan referensi abstraktif dan *F1 – Score* 84.10 untuk *ROUGE – 1*, 83.10 untuk *ROUGE – 2*, dan 83.31 untuk *ROUGE – L* dengan referensi ekstraktif.
- Pembuatan struktur *neural network* dan berbagai pengujian yang dilakukan jauh lebih cepat dengan menggunakan *GPU* yang memadai.
- Aplikasi berbasis *web* yang dibuat untuk mencoba model terdiri dari 2 halaman yaitu halaman input dan hasil. Sifatnya adalah local sehingga tidak bisa diakses dari luar.

5.2 Saran

Saran yang diberikan untuk penyempurnaan dan pengembangan lebih lanjut untuk penelitian ini adalah sebagai berikut:

- Melakukan *tuning* dari parameter – parameter yang ada sehingga dapat ditemukan parameter yang optimal untuk meningkatkan peforma model.
- Dilakukan *preprocessing* dengan menghilangkan tanda baca dan tanda special lainnya.
- Mencoba menggunakan model untuk melakukan pembuatan ringkasan secara abstraktif.

6. DAFTAR PUSTAKA

[1] Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E., Kochut, K., & Juan, G. 2017. Text Summarization Techniques: A Brief Survey. *Computation and Language*.

[2] Anagnostopoulos, A., Broder, A., Gabrilovich, E., Josifovski, V., & Riedel, L. 2011. Web page summarization for just-in-time contextual advertising. *ACM Transactions on Intelligent Systems and Technology*, 1-32.

[3] Cho, K., Merrienboer, B. v., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv preprint arXiv:1406.1078*,.

[4] Gunawan, D., Pasaribu, A., Rahmat, R. F., & Budiarto, R. 2017. Automatic Text Summarization for Indonesian Language. *IOP Conference Series: Materials Science and Engineering*, 190-196.

[5] Kurniawan, K., & Louvan, S. 2018. IndoSum: A New Benchmark Dataset for Indonesian Text Summarization. *2018 International Conference on Asian Language Processing (IALP)*, 215-220

[6] Nallapati, R., Zhai, F., & Zhou, B. 2016. SummaRuNNer: A Recurrent Neural Network based Sequence Model for. *Computation and Language*, 3075-3081.

[7] Radef, D. R., Hovy, E., & McKeown, K. 2002. Introduction to the Special Issue on Summarization. *Computation Linguistics*, 399-408.