Named Entity Recognition pada Resep Makanan dengan Metode *Bidirectional Long Short-Term Memory dan Bidirectional Encoders Representations from Transformers*



Disusun Oleh:

N a m a : Febrianto Eko Saputro

NIM : 17523165

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2022

HALAMAN PENGESAHAN DOSEN PEMBIMBING

Named Entity Recognition pada Resep Makanan dengan Metode *Bidirectional Long Short-Term Memory dan Bidirectional Encoders Representations from Transformers*

TUGAS AKHIR

Disusun Oleh:

N a m a : Febrianto Eko Saputro

NIM : 17523165

Yogyakarta, 27 Desember 2021

Pembimbing,

(Arrie Kurniawardhani, S.SI., M. Kom.)

HALAMAN PENGESAHAN DOSEN PENGUJI

Named Entity Recognition pada Resep Makanan dengan Metode *Bidirectional Long Short-Term Memory* dan *Bidirectional Encoders Representations from Transformers*

TUGAS AKHIR

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 27 Desember 2021

Tim Penguji

Arrie Kurniawardhani, S.SI., M. Kom. _____

Anggota 1

Ahmad Luthfi, S.Kom., M.Kom. _____

Anggota 2

Elyza Gustri Wahyuni, S.T., M.Cs. _____

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Febrianto Eko Saputro

NIM : 17523165

Tugas akhir dengan judul:

Named Entity Recognition* pada Resep Makanan dengan Metode *Bidirectional Long Short-Term Memory* dan *Bidirectional Encoders Representations from Transformers

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 27 Desember 2021



(Febrianto Eko Saputro)

HALAMAN PERSEMBAHAN

Saya persembahkan tugas akhir ini untuk kedua orang tua dan keluarga saya, yang telah memberikan semangat, yang telah mendidik dan mengasuh hingga saat ini serta teman-teman dan seluruh pihak yang telah memberikan semangat, bantuan dan motivasi selama proses pengerjaan dari awal hingga akhir.

HALAMAN MOTO

"Berlomba lomba dalam kebaikan"
-Q.S. Al-Baqarah Ayat 148

"*Be who you are and say what you feel because those who mind don't matter and those whomatter don't mind.*"
- Dr. Seuss

KATA PENGANTAR

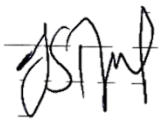
Assalamualaikum Warahmatullahi Wabarakatuh.

Alhamdulillahirabbil 'alamin, penulis memanjatkan puji syukur kepada Allah S.W.T. yang telah memberikan rahmat, hidayah, nikmat serta karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir mengenai "*NAMED ENTITY RECOGNITION* PADA RESEP MAKANAN DENGAN METODE *BIDIRECTIONAL LONG SHORT-TERM MEMORY DAN BIDIRECTIONAL ENCODERS REPRESENTATIONS FROM TRANSFORMERS*" guna memenuhi syarat untuk menyelesaikan pendidikan pada jenjang sarjana. Dalam proses penyelesaiannya, tentu tidak lepas dari beberapa kendala yang dihadapi, namun, atas do'a, bimbingan, dukungan, semangat serta motivasi dari segala pihak dan diri sendiri, satu per satu kendala tersebut berhasil dilalui. Oleh karena itu, penulis ingin menyampaikan apresiasi dan rasa terima kasih kepada

1. Orang tua penlulis yang selalu ada, selalu kasih semangat, selalu kasih saran, mendoakan penlulis.
2. Seluruh keluarga yang selalu memberi motivasi dan mendoakan penlulis.
3. Bapak Prof. Fathul Wahid, S.T., M.Sc., Ph.D. selaku Rektor Universitas Islam Indonesia.
4. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc. selaku Kaprodi Informatika – Program Sarjana UII.
5. Bapak Ahmad Fathan Hidayatullah, S.T., M.CS. dan Ibu Arrie Kurniawardhani, S.SI., M.Kom. selaku dosen pembimbing yang sudah bersedia meluangkan waktu dan tenaga dalam memberikan bimbingan dan ilmu kepada penulis dalam menyelesaikan tugas akhir dengan baik.
6. Seluruh dosen Informatika UII yang telah memberikan ilmunya yang sangat bermanfaat, semoga bapak dan ibu selalu diberi kesehatan dan lindungan dari Allah SWT.
7. Teman-teman Halghib yang senantiasa selalu menemani serta memberikan semangat positif, motivasi dan dukungan dalam proses mejalankan penelitian.
8. Kepeda semua orang yang selalu menbantu, mendoakan dan memberikan semangatkepada penulis.

Penulis menyadari bahwa tugas akhir ini jauh dari kata sempurna dan masih memiliki kekurangan, namun penulis berharap dengan disusunnya laporan tugas akhir ini bisa bermanfaat untuk semua orang serta dapat dikembangkan menjadi hal yang lebih baik lagi ke depannya yang dapat berguna bagi semua orang

Yogyakarta, 27 Desember 2021



(Febrianto Eko Saputro)

SARI

Dalam kehidupan kita sehari-hari makanan merupkan hal yang sangat penting. Resep Masakan merupakan panduan bagi seseorang dalam mengolah bahan makanan menjadi hidangan yang dapat disajikan. Namun sebagian orang menanggap bagian bahan dalam resep masakan tidak penting. Informasi bahan resep masakan sendiri dapat membantu orang dalam beberapa skenario seperti estimasi biaya dalam belanja dan estimasi nutrisi dalam makanan.

Named Entity Recognition (NER) dapat digunakan untuk menemukan informasi penting yang diinginkan dengan lebih mudah dengan mendeteksi entitas pada suatu teks. Tujuan penelitian ini adalah membuat model NER untuk membantu mengidentifikasi entitas pada resep masakan dan mengetahui performa model yang dibangun. Terdapat 3999 resep masakan yang didapatkan dari website *Kaggle.* Resep yang terkumpul akan dipecah menjadi 196.696 token kata. Kemudian token- token tersebut akan dilabeli sesuai dengan kategorinya, yaitu nama bahan makanan, bumbu masakan dan alat. Selain kategori yang sudah disebutkan, maka token akan diberi label O (*outside*).

Untuk mencapai tujuan penelitian, peneliti menggunakan lima langkah penelitian, yaitu mengumpulkan *dataset*, *preprocessing*, ekstraksi fitur dengan *word embedding* berupa *Word2Vec*, pemodelan NER, evaluasi dan deteksi entitas serta menggunakan metode *Bidirectional Long Short-Term Memory* (BiLSTM) dan *Bidirectional Encoder Representaionsfrom Transformers* (BERT) pada pemodelannya. Dari hasil yang telah diperoleh, model yang dihasilkan mampu memprediksi entitas dengan baik, yaitu menghasilkan 77,9% untuk model BiLSTM dan 99,4% untuk model BERT. Namun, masih ditemukan beberapa kesalahan dalam mendeteksi entitas, seperti misalnya model masih salah dalam mengidentifikasi entitas yang harusnya dilabeli bahan tetapi justru dilabeli bumbu. Dari skenario-skenario model yang telah diuji, diperoleh model BERT yang memiliki rata-rata F1-*Score* tertinggi sebesar 94,4%, dibandingkan model BiLSTM yang memiliki rata-rata F1-*Score* tertinggi sebesar 77,9%.

Kata kunci: BiLSTM, BERT, *Named entity recognition*, resep makanan, *Word2Vec*

GLOSARIUM

Bobot	parameter pada *neural network* yang mentransformasikan data masukan dalam lapisan tersembunyi pada jaringan.
Dataset	himpunan data yang digunakan ketika membangun fitur dan model.
Hyperparameter	variabel yang mempengaruhi keluaran model
List	tipe data pada *python* yang mampu menyimpan lebih dari satu daftar nilai
Node	titik-titik yang menghubungkan satu jaringan ke jaringan lainnya.
Parameter	variabel konfigurasi yang bersifat internal pada model.
Tuple	tipe data pada *python* yang digunakan untuk menyimpan sekumpulan data yang isinya tidak dapat berubah
Overfitting	*Overfitting* terjadi ketika model menjadi sangat baik dalam mengklasifikasikan atau memprediksi data.
Hidden Layer	lapisan tersembunyi.
Dropout	teknik regulasi untuk mengurangi *overfitting.*

DAFTAR ISI

HALAMAN JUDUL i
HALAMAN PENGESAHAN DOSEN PEMBIMBING ii
HALAMAN PENGESAHAN DOSEN PENGUJI iii
HALAMAN PERSEMBAHAN v
HALAMAN MOTO vi
KATA PENGANTAR vii
SARI ix
GLOSARIUM x
DAFTAR ISI xi
DAFTAR TABEL xiii
DAFTAR GAMBAR xiv
BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Tujuan Penelitian 2
1.4 Batasan Masalah 2
1.5 Manfaat Penelitian 3
1.6 Sistematika Penelitian 3
BAB II LANDASAN TEORI 4
2.1 Penelitian Terkait 4
2.2 Dasar Teori 7
2.2.1 Natural Language Processing (NLP) 7
2.2.2 Named Entity Recognition 7
2.2.3 LSTM 8
2.2.4 BiLSTM 9
2.2.5 BERT 10
2.2.6 Ektraksi Fitur 12
2.2.7 Word2Vec 12
BAB III METODOLOGI PENELITIAN 14
3.1 Tahap-tahap Pengerjaan Tugas Akhir 14
3.2 Uraian Metodologi 14
3.2.1 Pengambilan *Dataset* 14
3.2.2 Preprocessing 15
3.2.3 Pelabelan Entitas 17
3.2.4 Ekstraksi Fitur 19
3.2.5 Pemodelan Named Entitty Recognition 20
3.2.6 Evaluasi Model 24
3.2.7 Deteksi Entitas 25
BAB IV HASIL DAN PEMBAHASAN 26
4.1 Dataset 26
4.2 Preprocessing 26
4.3 Pelabelan Entitas 28
4.4 Mempersiapkan Data Masukan 29
4.5 Ekstraksi Fitur 32
4.6 Pemodelan Named Entity Recognition 32
4.7 Evaluasi Model 36
4.8 Deteksi Entitas 40

BAB V KESIMPULAN DAN SARAN 43
5.1 Kesimpulan 43
5.2 Saran.....43
DAFTAR PUSTAKA.....45
LAMPIRAN 47

DAFTAR TABEL

Tabel 2.1 Penelitian-Penelitian terkait 4
Tabel 3.1 Contoh menghilangkan karakter non ascii 15
Tabel 3.2 Contoh menghapus simbol 16
Tabel 3.3 Contoh menghapus spasi berlebih 17
Tabel 3.4 Contoh tokenisasi 17
Tabel 3.5 Jenis macam label dan penjelasan 18
Tabel 3.6 Contoh *Word2Vec* 19
Tabel 3.7 Rangkuman scenario model BiLSTM 23
Tabel 3.8 Rangkuman scenario model BERT 24
Tabel 4.1 Jumlah token pada setiap entitas 29
Tabel 4.2 Hasil evaluasi skenario 1 model BiLSTM 37
Tabel 4.3 Hasil evaluasi skenario 2 model BiLSTM 37
Tabel 4.4 Hasil evaluasi skenario 3 model BiLSTM 38
Tabel 4.5 Hasil evaluasi skenario 1 model BERT 39
Tabel 4.6 Hasil evaluasi skenario 2 model BERT 39
Tabel 4.7 Hasil evaluasi skenario 3 model BERT 40

DAFTAR GAMBAR

Gambar 2.1 LSTM memory cell 8
Gambar 2.2 Arsitektur BiLSTM 9
Gambar 2.3 Perbedaan antara BERT BASE dan BERT LARGE. 10
Gambar 2.4 Arsitektur BERT 11
Gambar 2.5 Arsitektur CBOW 13
Gambar 3.1 Langkah-langkah pengerjaan tugas akhir 14
Gambar 3.2 *Dataset* NER 15
Gambar 3.3 Teknik dropout 22
Gambar 4.1 Kode program menghapus non-ascii 26
Gambar 4.2 Kode program untuk menghapus symbol 27
Gambar 4.3 Kode program menghapus spasi berlebih 27
Gambar 4.4 Hasil *preprocessing* 27
Gambar 4.5 Kode program tokenisasi 27
Gambar 4.6 Hasil tokenisasi 28
Gambar 4.7 Hasil pelabelan entitas 29
Gambar 4.8 Kode program untuk persiapan masuk data ke model BilSTM 30
Gambar 4.9 Kode program untuk persiapan data masuk ke kelas *Word2Vec* 31
Gambar 4.10 Kode program persiapan data masuk ke model BERT 32
Gambar 4.11 Kode program *Word2Vec* 32
Gambar 4.12 Kode program *splitting data* model BiLSTM 33
Gambar 4.13 Kode program *splitting data* model BERT 33
Gambar 4.14 Kode program membangun model BiLSTM 34
Gambar 4.15 Kode program membangun model BERT 34
Gambar 4.16 Kode program melatih model BiLSTM 35
Gambar 4.17 Kode program melatih model BERT 36
Gambar 4.18 Hasil deteksi entitas model BiLSTM 41
Gambar 4.19 Hasil deteksi entitas model BERT 41

BAB I
PENDAHULUAN

1.1 Latar Belakang

Resep Masakan merupakan panduan bagi seseorang dalam mengolah bahan makanan menjadi hidangan yang dapat disajikan. Resep masakan diperlukan dalam pengolahan makanan untuk rumah tangga maupun usaha seperti misalnya katering dan restoran. Namun sebagian orang menanggap bagian bahan dalam resep masakan tidak penting. Informasi bahan resep masakan sendiri dapat membantu orang dalam beberapa skenario seperti estimasi biaya dalam belanja dan estimasi nutrisi dalam makanan (Diwan et al., 2020).

Oleh karena itu, diperlukan *Named Entity Recognition* (NER) yang bisa mendapatkan informasi penting seperti bumbu dan bahan makanan dengan lebih mudah. *Named Entity Recognition* (NER) dipilih karena dapat digunakan untuk mengidentifikasi bagian teks yang dianggap sebagai informasi penting seperti bahan makanan dan bumbu, sehingga informasi penting yang diinginkan dapat didapatkan dengan lebih mudah (Wibisono & Khodra, 2018). *Named Entity Recognition* adalah proses otomatis mengekstraksi entitas bernama yang dianggap penting di dalam teks dan menentukan kategorinya ke dalam kategori terdefinisi (Wibisono & Khodra, 2018).

Penelitian ini bertujuan untuk melakukan identifikasi yang memuat informasi tentang hal apa saja yang ada di resep makanan. Informasi yang dimaksud yaitu entitas bahan, bumbu, dan alat. Maka dari itu, penelitian ini juga bertujuan menguji model yang digunakan dalam mengidentifikasi resep masakan. Selain itu, penelitian ini akan memberikan kontribusi terhadap NER pada domain masakan dalam bahasa Indonesia, karena domain masakan terhadap NER masih terbatas. NER pada domain masakan ini merupakan langkah pertama untuk topik penelitian terkait ekstraksi informasi. Selain itu, penelitian ini juga membantu orang yang ingin belajar memasak untuk mengidentifikasi bahan-bahan dalam resep makanan.

Penelitian ini akan menggunakan metode *Bidirectional Long Short-Term Memory* (BiLSTM) dan *Bidirectional Encoders Representations from Transformer*

(BERT) dalam membangun model NER. Penelitian menggunakan metode BiLSTM sebelumnya pernah dilakukan oleh (Mahalakshmi et al., 2020). Penelitian tersebut menunjukkan bahwa metode BiLSTM dapat membangun model NER dengan baik karena dapat mempelajari informasi dari masa lalu dan informasi dari masa depan, sehingga model dapat memprediksi kata saat ini dengan baik. Pada penelitian NER menggunakan metode BERT pernah dilakukan (Chantrapornchai & Tunsakul, 2021). Metode BERT merupakan metode terkini yang dapat digunakan untuk membangun model dan memiliki hasil yang baik, seperti yang dilakukan pada penelitian(Chantrapornchai & Tunsakul, 2021).

1.2 Rumusan Masalah

Berdasarkan dari latar belakang yang telah dijelaskan, maka dapat dirumuskan suatu permasalahan:

a. Bagaimana membangun model *Named Entity Recognition* (NER) yang dapat mengidentifikasi entitas pada resep makanan menggunakan metode BiLSTM dan BERT.
b. Bagaimana performa model NER menggunakan metode BiLSTM dan BERT yang dihitung menggunakan F1-*Score* dalam mengidentifikasi entitas pada resep makanan.

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

a. Membangun model *Named Entity Recognition* (NER) yang dapat mengidentifikasi entitas pada resep makanan menggunakan metode BiLSTM dan BERT.
b. Menguji performa model BiLSTM dan BERT dalam mengidentifikasi entitas pada resep makanan dengan F1-*Score.*

1.4 Batasan Masalah

Batasan-batasan yang terdapat dalam penelitina ini yaitu:

a. Dataset yang digunakan diambil dari *Kaggle*.
b. Data yang digunakan hanya data yang berbahasa Indonesia saja.
c. Entitas yang diidentifikasi hanya bahan, bumbu dan alat.
d. Tidak bisa menangani data Bahasa Indonesia yang tidak baku.

1.5 Manfaat Penelitian

Manfaat penelitian ini dapat mengetahui dan mengidentifikasi bahan-bahan dalam resep masakan sehingga dapat mempermudah dalam mengenali bahan, bumbu dan alat.

1.6 Sistematika Penelitian

Sistematika penulisan dalam laporan tugas akhir sebagai berikut:

BAB I Pendahuluan

Merupakan bab yang berisi tentang gambaran umum, latar belakang penelitian tentang "Named Entity Recognition pada Resep Makanan Menggunakan Metode Bidirectional Short-Term Memory dan Bidirectional Encoders Representations from Transformers (BERT)", rumusan masalah, tujuan, batasan penelitian, manfaat penelitiandan sistematika penulisan.

BAB II Landasan Teori

Merupakan bab yang berisi penjelasan mengenai penelitian yang telah dilakukan oleh penelitian sebelumnya dan juga dasar teori penelitian *Named Entity Recognition.*

BAB III Metodologi Penelitian

Memberikan penjelasan tentang tahapan-tahapan yang digunakan dalam penelitian *Named Entity Recognition* ini.

BAB IV Hasil dan Pembahasan

Berisi penjelasan mengenai hasil dari penelitian yang telah dilakukan serta evaluasi terhadap penelitian yang dilakukan.

BAB V Kesimpulan dan Saran

Merupakan bab yang berisi tentang kesimpulan yang didapatkan dari penelitian yang telah dilakukan serta saran yang dapat digunakan untuk penelitian selanjutnya.

BAB II
LANDASAN TEORI

2.1 Penelitian Terkait

Bagian ini akan menjelaskan tentang penelitian yang dilakukan pada penelitian sebelumnya dan dijadikan sebagai referensi dalam membuat penelitian ini. Dalam proses pencarian penlitian ada beberapa tahapan yang dilakukan. Pertama, mencari penelitian melalui *Google Scholar* dengan batasan tahun dari tahun 2017 hingga 2021. Kedua, kata kunci yang digunakan untuk mencari penelitian adalah "Named Entity Recognition for food ingredients". Pada Tabel 2.1, dapat dijelaskan untuk penelitian yang telah di dapatkan menggunakan Bahasa Inggris dan metode yang digunakan dalam penelitian tersebut menggunakan *deep learning* dan *machine learning*. Penelitian- penelitian tersebut ditunjukkan pada Tabel 2.1.

Tabel 2.1 Penelitian-Penelitian terkait

No	Judul Penelitian	Metode	Hasil
1	*Exploring Bi-LSTMs for Named Entity Recognition in Indian Culinary Sciene* (Mahalakshmi et al., 2020)	BiLSTM, CNN dan CRF	Penelitian ini menerapkan NER pada resep makanan India dengan menggunakan metode BiLSTM-CNN. Dataset diambil dari website *Hebber's Kitchen.* Nilai F1 score tanpa *pre-processing* mendapatkan nilai 92.87%dan nilai F1 score setelah di *pre- processing* mendapatkan nilai 94.66%
2	*Named Entity Recognition for Indonesian Language using Bidirectional LSTM-CNNs* (Gunawan et al., 2018)	BiLSTM dan CNN	Penelitian ini mengimplementasikan NER untuk Bahasa Indonesia menggunakan pendekatan deep learning BiLSTM dan CNN dengan mengekstraki informasi dari artikel- artikel tersebut kemudian akan dibedakan ke dalam 4 kelas yang berbeda yaitu: *Person, Organization, Location,* dan *Event.* Gabungan BiLSTM dan CNN mendapatkan nilai *Precision* 79.72%, nilai *Recall* 79.61%

			dan *F-Score* 79.43%
3	*A Semi-Supervised BERT Approach for Arabic Named Entity Recognition* (Helwe et al., 2020)	BERT	Penelitian ini mengimplementasikan NER untuk Bahasa Arab menggunakan pendekatan deep learning BERT. Mereka membandingkan pendekatan nya terhadap berbagai dasar dan tools NER Arab pada tiga *datasets*: AQMAR, NEWS, TWEETS. Pada *datasets* AQMAR dan NEWS mereka mendapatkan peningkatan signifikandalam *F-measure*, yang ditulis dalam Bahasa Arab Standar Modern, dan pada *dataset* TWEETS mereka
4	*Beheshti-NER: Persian named entity recognition Using BERT* (Taher et al., 2019)	BERT	Penelitian ini mengimplementasikan NER untuk Bahasa Persia menggunakan pendekatan deep learning BERT. Mereka menggunakandua *datasets* untuk melatih dan menguji model mereka yaitu: PEYMA dan ARMAN. Pada *dataset* PEYMA mereka mendapatkna F1 score 90.59% di *pharse-level* dan 87.62% di *word-level*. Kelas *Person* mendapatkan hasil terbaik dan kelas *Time* mendapatkan hasil terburk. Pada *dataset* ARMAN mereka mendapatkan F1 score 79.93% di *phrase-level* dan 84.03% di *word- level.* Kelas *Person* mendapatkan hasil terbaik dan

			kelas *Event* mendapatkan hasil terburuk.
5	*A Named Entity Based Approach to Model Recipes* (Diwan et al., 2020)	Stanford NER Model, *K-means clustering*.	Penelitian ini mengimplementasikan NER untuk resep makanan menggunakan model Stanford NER dengan *K-means clustering.* Mereka menggunakan satu *dataset* yaitu RecipeDB. Model mereka mendapatkan F1 score 0.95 di semua *dataset.*

Penelitian yang akan dilakukan oleh peneliti adalah *named entity recognition* (NER) pada domain masakan menggunakan metode BiLSTM dan BERT. NER dilakukan pada bahan-bahan makanan untuk memperoleh kata-kata kunci pada resep masakan. Dalam pelabelam sekuensial seperti NER, model yang dapat memilki informasi dari masa lalu dan masa depan akan berguna, oleh karena itu dibutuhkan BiLSTM dalam NER. BiLSTM dipilih karena untuk mengidentifikasi entitas membutuhkan pemahaman yang baik terhadap suata konteks kata dalam kalimat. BiLSTM memiliki dua lapisan yang bergerak maju dan mundur. Lapisan yang bergerak maju akan bergerak dari awal kalimat hingga akhir kalimat. Lapisan yang bergerak mundur akan bergerak dari akhir kalimat hingga awal kalimat sehingga dapat memperoleh informasi dari kedua arah. Oleh karena itu, BiLSTM dipilih karena dapat melakukan pemahaman dengan memperhatikan konteks dari kata sebelumnya dan sesudahnya (X. Ma & Hovy, 2016). BERT dipilih karena BERT merupakan *language representations model* yang dapat meningkatkan tugas NLP yang bertujuan untuk memprediksi hubungan antar kata dengan

menganalisisnya secara keseluruhan seperti NER (Souza et al., 2019). BERT dirancang untuk mempelajari teks tidak berlabel dari sisi kiri dan kanan dengan mengkondisikan konteks di semua lapisan dan mempelajari hubungan kontekstual antara kata-kata dalam teks. BERT memiliki keunggulan dengan memiliki Transformer. Transformer merupakan mekanisme yang mempelajari hubungan kontekstual antara kata-kata dalam teks (Fimoza, 2021). Oleh karena itu, BERT akan digunakan dalam penelitian ini. Dataset yang digunakan merupakan data berbahasa Indonesa dari website *Kaggle* yang berjudul *Indonesian food recipes*. Ektraksi fitur yang digunakan dalam penelitian ini adalah *word embedding* dengan Teknik *Word2Vec*.

2.2 Dasar Teori

2.2.1 Natural Language Processing (NLP)

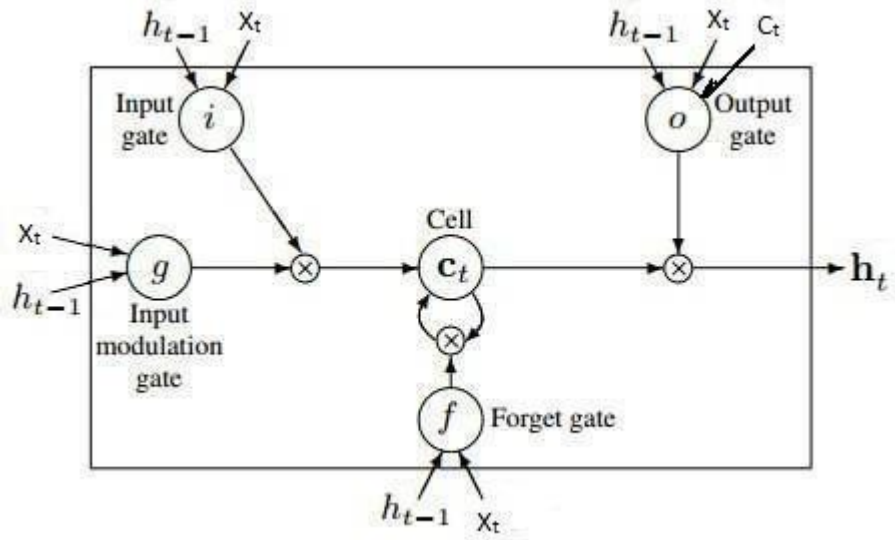
Natural Language Processing (NLP) menyelidiki penggunaan komputer untuk memproses atau memahami bahasa manusia untuk tujuan melakukan tugas yang bermanfaat. NLP adalah bidang yang menggabungkan linguistik komputasi, ilmu komputasi, dan kecerdasan buatan. Aplikasi khas di NLP termasuk pengenalan suara, pemahaman bahasa lisan, sistem dialog, analisis leksikal, parsing, *machine translation*, grafik pengetahuan, pencarian informasi, menjawab pertanyaan, analisis sentimen, komputasi sosial, generasi bahasa alami dan ringkasan bahasa alami (Deng & Liu, 2018). *Natural Language Processing* difokuskan pada desain dan analisis algoritma komputasi dan representasi untuk memproses bahasa manusia alami. Tujuan NLP adalah untuk menyediakan kemampuan komputasi baru seputar bahasa manusia: misalnya, mengekstrak informasi dari teks (Eisenstein, 2019).

2.2.2 Named Entity Recognition

Named Entity Recognition (NER) adalah submasalah dari ekstraksi informasi dan melibatkan pemrosesan terstruktur dan dokumen tidak terstruktur dan mengidentifikasi ekspresi yang merujuk pada orang, tempat, organisasi, dan perusahaan. NER adalah tugas mendasar dan merupakan inti dari sistem *Natural Language Processing* (NLP). NER melibatkan dua tugas, yang pertama adalah identifikasi nama yang tepat di teks, dan kedua klasifikasi nama-nama ke dalam satu set kategori yang telah ditentukan, seperti orang nama, organisasi (perusahaan, pemerintah organisasi, komite, dll), lokasi (kota, negara, sungai, dll), tanggal dan waktu (Mansouri et al., 2008).

2.2.3 LSTM

Long Short-Term Memory (LSTM) adalah arsitektur RNN yang dimodifikasi dengan penambahan *memory cell* yang dapat menyelesaikan masalah *vanishing gradient* dan masalah proses data sekuensial yang panjang dengan menggunakan suatu set gerbang yang digunakan untuk mengontrol ketika informasi masuk ke dalam memori (Manaswi et al., 2018).



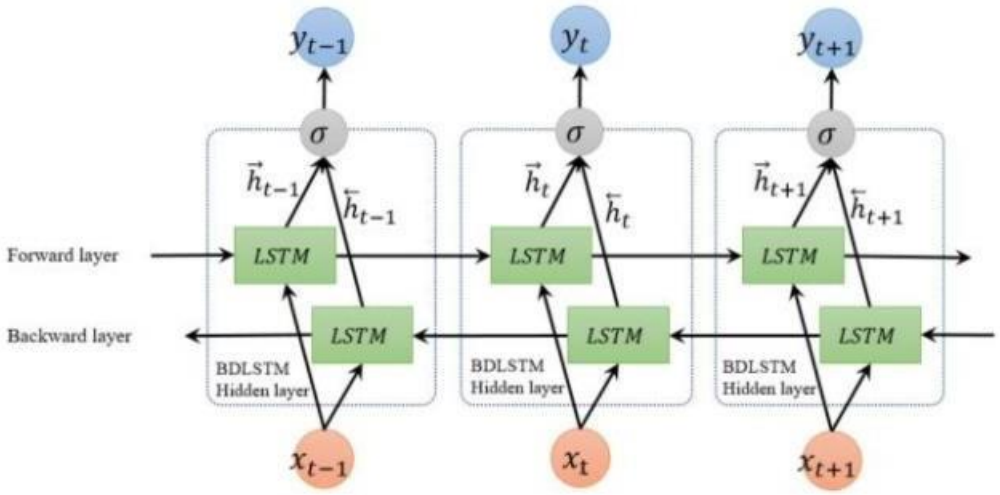
Gambar 2.1 LSTM memory cell

Sumber: *pydeeplearning.weebly.com*

Di dalam LSTM *memory cell* terdiri dari tiga *gate*, yaitu *input gate*, *forget gate* dan *output gate. Input gate* adalah *gate* yang megubah nilai *cell state. Forget gate* adalah *gate* yang digunakan untuk menghapus informasi dari *cell state.* Informasi dari *hidden state* (h_{t-1}) sebelumnya dan masukan saat ini (X_t) akan dihitung oleh fungsi sigmoid dengan nilai keluaran antara 0 dan 1. Jika nilai keluarannya 0, maka informasi akan dihapus. Jika nilai keluarannya 1, maka informasi akan disimpan. *Output gate* adalah *gate* yang menentukan *hidden state* dan *cell state* yang baru (Permana & Purnamasari, 2019).

2.2.4 BiLSTM

Bidirectional Long Short-Term Memory (BiLSTM) merupakan adaptasi dari LSTM yang mempunyai dua lapisan LSTM, *forward* LSTM dan *backward* LSTM (Kågebäck & Salomonsson, 2016). *Forward* LSTM dan *Backward* LSTM dapat menggambil dan menggabukan informasi dari kalimat masa lalu dan masa depan, sehingga BiLSTM dapat memahami konteks dengan lebih baik dari LSTM. Gambar 2.2 menunjukkan arsitektur dari BiLSTM.



Gambar 2.2 Arsitektur BiLSTM

Sumber: *gabormelli.com*

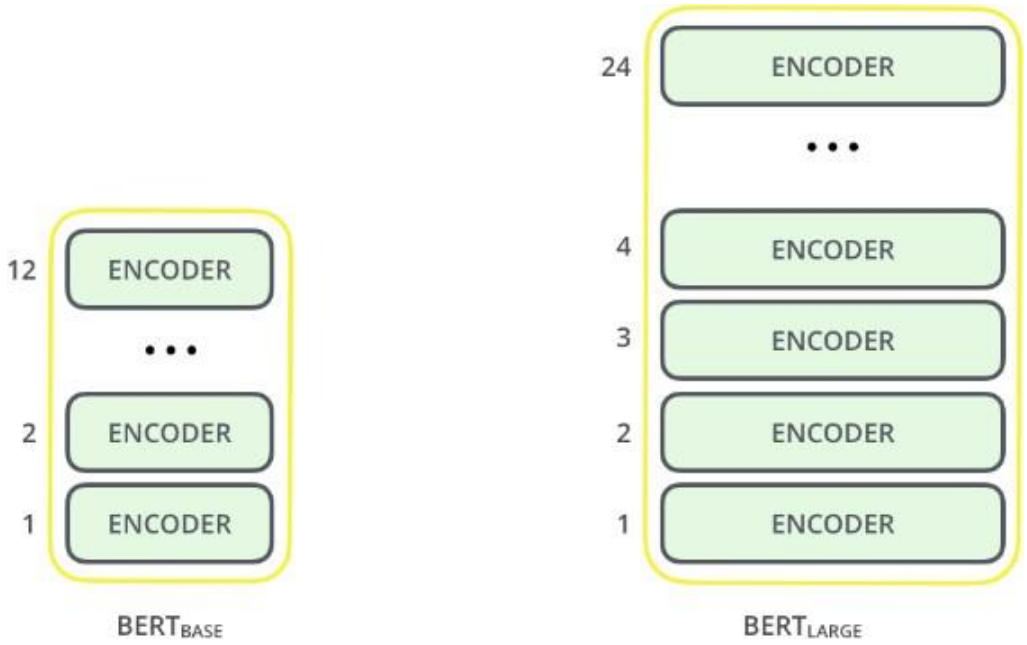
Dari arsitektur tersebut, terdapat dua hidden *state* yaitu $\vec{h}$ yang mempresentasikan *forward layer* yang mengambil kalimat dari masa lalu dan $h^{\leftarrow}$ yang mempresentasikan *backward layer* yang mengambil kalimat dari masa depan. Kedua lapisan tersebut digabungkan dengan fungsi σ untuk mendaptkan satu keluaran yaitu y_t. *Forward layer* dihitung secara berulang dengan masukan dari $t-1$ ke $t+1$. Sedangkan *backward layer* dihitung secara berulang dengan masukan dari $t+1$ ke $t-1$.

BiLSTM sangat bermanfaat dalam pelabelan sekuensial karena pada *hidden state* BiLSTM memiliki akses informasi ke masa lalu dan masa depan. Sedangkan *hidden state* pada LSTM hanya memiliki akses ke masa lalu dan tidak bisa memiliki akses ke masa depan (X. Ma & Hovy, 2016).

2.2.5 BERT

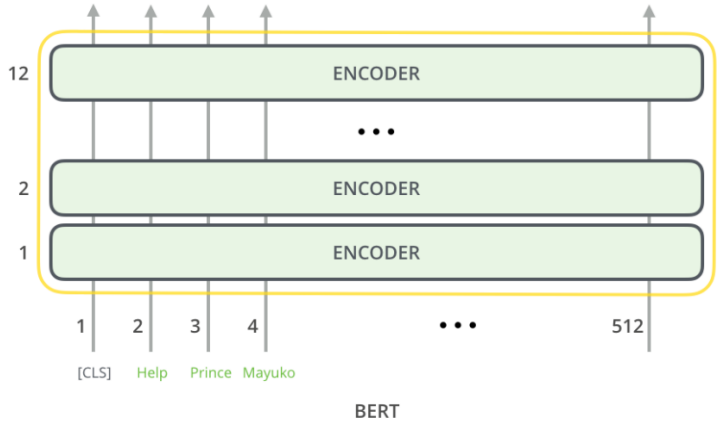
Bidirectional Encoder Representations Transformers (BERT) adalah *language representations model* yang dirancang untuk mempelajari teks tidak berlabel dari sisi kiri dan kanan dengan mengkondisikan konteks di semua lapisan (Devlin et al., 2019). Sesuai dengan namanya, BERT menggunakan Transformer. Transformer adalah sebuah mekanisme yang mempelajari hubungan kontekstual antara kata-kata dalam teks. Transformer dapat memahami dan mengkonversi kata-kata dalam teks dengan menggunakan *self-attention* (Fimoza, 2021).

Arsitektur model BERT berupa *multi-layer bidirectional* Transformer encoder. Terdapat dua ukuran model yang ada pada BERT, yaitu $BERT_{Base}$ dan $BERT_{Large}$. Kedua model ini memiliki banyak Transformer Blocks. Model pertama, $BERT_{Base}$ memiliki 12 *layers, hidden size* sebesar 768, 12 *self-attention heads* dan 110M *parameters.* Model kedua, $BERT_{Large}$ memiliki 24 *layers, hidden size* sebesar 1024, 24 *self-attention heads* dan 340 *parameters.* Gambar 2.3 menunjukkan perbedaan antara $BERT_{Base}$ dan $BERT_{Large}$:



Gambar 2.3 Perbedaan antara BERT BASE dan BERT LARGE.

Sumber: *jalammar.github.io*



Gambar 2.4 Arsitektur BERT

Sumber: *jalammar.github.io*

BERT menggunakan WordPiece embedding dengan 30,000 token vocabulary. Di setiap urutan pertama ada token klasifikasi khusus yaitu [CLS] yang digunakan sebagai penggabung untuk tugas klasifikasi yang dimana dua kalimat digabungkan menjadi satu urutan. BERT ada dua cara untuk mengetahui kalimat. Pertama, memisahkan dengan token khusus yaitu [SEP]. Kedua, menambahkan learned embedding ke setiap token menunjukkan apakah token tersebut milik kalimat A atau kalimat B.

Ada dua langkah yang terlibat dalam BERT: *pre-training* dan *fine tuning*. Saat *pre-training*model BERT memahami dan mempelajari pada data tidak berlabel dengan dua tugas *unsupervised* yaitu *Masked Language Model* (MLM) dan *Next Sentence Prediction* (NSP).

a. Masked Language Model (MLM)

Tugas MLM adalah memberi *mask* atau menutupi 15% kata-kata ke setiap urutan token input secara acak dan kemudian memprediksi nilai asli dari token yang ditutupi.

b. Next Sentence Prediction (NSP)

Dalam proses *pre-training*, BERT dapat menerima pasangan kalimat sebagai input dan dilatih untuk memprediksi jika pasangan kalimat tersebut adalah kalimat berikutnya atau hanya satu kalimat saja. Jika ada kalimat A dan B, maka 50% BERT akan menganggap bahwa kalimat B adalah kalimat berikutnya dari kalimat A. Sedangkan 50% lainnya adalah kalimat acak yang diambil dari *corpus*.

Untuk melatih sebuah model bahasa, classifier perlu dilatih dengan sedikit perubahan padamodel BERT selama fase pelatihan (training) yang disebut finetuning. Menurut (Devlin et

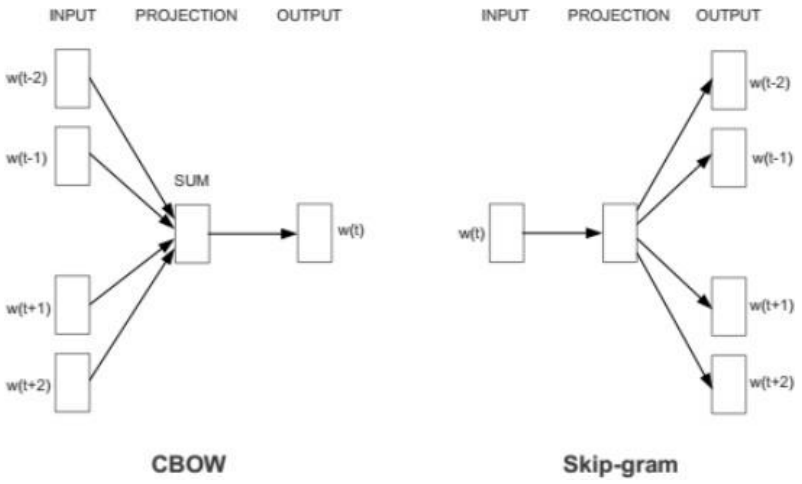
al., 2019) terdapat rekomendasi hyperparameters yang dapat di-fine-tuning untuk mencapai hasil yang maksimal. Fine-tuning sangat mudah dilakukan karena mekanisme self-attention di Transformer membuat BERT bisa membuat model untuk berbagai tugas,baik pada kalimat tunggal (single sentence) atau kalimat berpasangan, dengan menukar masukan dan keluaran yang sesuai.

2.2.6 Ektraksi Fitur

Ektraksi fitur merupakan proses data mentah menjadi karakteristik informatif, yang akan paling efisien atau bermakna untuk mewakili informasi penting untuk dianalisis atau klasifikasi (Elnemr et al., 2016). Tujuan ektraksi fitur adalah untuk memperoleh informasi yang berguna dari sampel asli dan mewakili informasi tersebut sebagai vektor yang dinormalisasi dengan ukuran yang sama. Ekstraksi fitur dapat membantu meningkatkankinerja sistem dalam memberikan prediksi (Huang et al., 2016). Jenis ekstraksi fitur ada berbagai macam, akan tetapi yang akan digunakan dalam penelitian ini adalah *word embedding* dengan teknik *Word2Vec*. *Word embedding* dipilih karena menurut penelitian (Seok et al., 2015) mampu membantu kinerja NLP lebih baik dengan mengelompokkan kata-kata yang mirip. *Word embedding* juga dapat menangkap berbagai dimensi makna daninformasi relevan dengan fitur dari kata-kata dalam vector.

2.2.7 Word2Vec

Word2Vec merupakan salah satu teknik dari *word embedding* yang diusulkan oleh (Mikolov et al., 2013) di 2013 untuk ekspresi kata termasuk arti dan konteks kata dalam dokumen dan mencakup dua algoritma pembelajaran, yaitu *continuous bag-of-word* (CBOW) dan *skip-gram* (Jang et al., 2019). *Word2Vec* dipilih karena terbukti pada penelitian (Seok et al., 2015) memiliki performa yang baik sebagai fitur. Gambar 2.5 menunjukkan arsitektur *Word2Vec.*



Gambar 2.5 Arsitektur CBOW

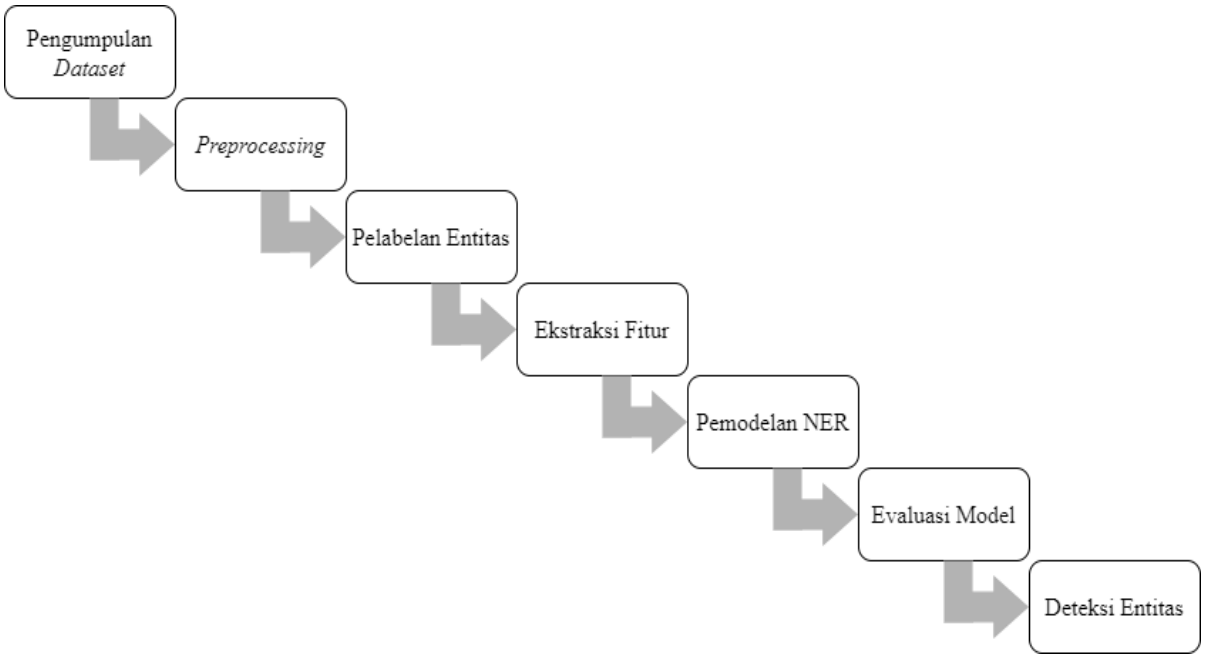
Sumber: (Mikolov et al., 2013)

Dalam *Word2Vec*, terdapat arsitektur *Continuous Bag-of-Words* (CBOW) dan *Skip-gram*.Arsitektur CBOW akan memprediksi kata saat ini dengan berdasarkan konteks kata di sekitarnya, sedangkan arsitektur *Skip-gram* menggunakan kata saat ini untuk memprediksi konteks sekitarnya (L. Ma & Zhang, 2015).

BAB III
METODOLOGI PENELITIAN

3.1 Tahap-tahap Pengerjaan Tugas Akhir

Gambar 3.1 menunjukan langkah-langkah yang digunakan dalam mengerjakan penelitian ini. Langkah-langkah tersebut terdiri dari pengumpulan *dataset*, *preprocessing*, pelabelan entitas, ektraksi fitur, pemodelan NER, evaluasi model dan deteksi entitas. Alur metodologi penelitian ini sama seperti penelitian (Permana & Purnamasari, 2019), di mana alur metodologi pada penelitian tersebut dijelaskan sesuai dengan langkah-langkah yang dilakukan secara urut. Oleh karena itu, metodologi pada penelitian ini juga dijelaskan secara urut sesuai dengan langkah yang dilakukan.



Gambar 3.1 Langkah-langkah pengerjaan tugas akhir

3.2 Uraian Metodologi

3.2.1 Pengambilan *Dataset*

Data yang digunakan dalam penelitian *Named Entity Recognition* ini merupakan dataset yang didapat dari *website* Kaggle yang berjudul *Indonesian Food Recipies.* Dataset yang digunakan tersebut menggunakan data kumpulan resep makanan yang diambil dari *website* Cookpad. Data tersebut dikategorikan dalam tujuh kategori, *dataset* ayam, *dataset* kambing,

dataset sapi, *dataset* telur, *dataset* tahu, *dataset* ikan dan *dataset* tempe. Di setiap kategori ada 5 kolom yaitu *Title*, *Ingredients, Steps, Love* dan *URL*. Peneliti hanya mengambil data dari dua kolom yaitu kolom *Title* dan *Ingredients.* Data yang dikumpulkan data berbahasa Indonesia. Data yang didapatkan dari *website* Kaggle merupakan data dalam bentuk file berekstensi .csv. Data yang digunakan dalam penelitian ini berjumlah 3999 resep masakan. Data tersebut akan digunakan sebagai data latih dan data uji unutk pemodelan dan mengujikinerja model. Gambar 3.2 menunjukkan data yang digunakan dalam penelitian ini.



Gambar 3.2 *Dataset* NER

3.2.2 Preprocessing

Sebelum membuat model akan ada tahap *preprocessing.* Dalam tahap *preprocessing* ini dilakukan untuk mempersiapkan data agar dapat diproses dan data masukan yang akan nantinya akan diproses menjadi mudah dimengerti oleh computer. Dalam penelitian ini *preprocessing* yang akan dilakukan yaitu menghapus karakter NON-ASCII, menghapus simbol, menghilangkan spasi berlebih dan tokenisasi.

a. Menghilangkan karakter NON-ASCII

Tahap ini digunakan untuk menghilangkan karakter-karakter NON-ASCII yang terdapat pada data. Karakter NON-ASCII merupakan karakter yang tidak termasuk dalam kategori kodeASCII. Contoh menghapus karakter ASCII ditunjukkan pada Tabel 3.1.

Tabel 3.1 Contoh menghilangkan karakter non ascii

Sebelum	**Sesudah**

1 piring nasi--Â¹/â• ´ fillet ayam, potong kotak, cuci bersih--Daun pisang yangsudah dibersihkan--Lidi/tusuk gigi untuk mengapit daun--1 lbr daun salam--3 cm serai, memarkan--1 ikat daun kemangi, opsional--1 cabe merah besar, iris tipis--1 sdt kecap manis--Minyak untuk menumis--100 ml air--Bumbu Halus:--1 cabe merah besar--3 cabe kecil--2 siung bawang merah--1 siung bawang putih--1 kemiri--1 cm kunyit--Â¹/Â² ruas jahe--Â¹/â• ´ tomat --1 sdt gula pasir--1 blok kaldu ayam (aku biasa pake maggy) pake royco, masako--Merica (dikira2 z takar nya, aku sdkit z sih) --	1 piring nasi--/ fillet ayam, potong kotak, cuci bersih--Daun pisang yang sudah dibersihkan--Lidi/tusuk gigi untuk mengapit daun--1 lbr daun salam--3 cm serai, memarkan--1 ikat daun kemangi, opsional--1 cabe merah besar, iris tipis--1 sdt kecap manis--Minyak untuk menumis--100 ml air--Bumbu Halus:--1 cabe merah besar--3 cabe kecil--2 siung bawang merah--1 siung bawang putih--1 kemiri--1 cm kunyit--/ ruas jahe--/ tomat--1 sdt gula pasir--1 blok kaldu ayam (aku biasa pake maggy) pake royco, masako--Merica (dikira2 z takar nya, aku sdkit z sih)--

b. Menghapus simbol

Tahap ini digunakan untuk menghapus simbol yang terdapat pada data. Tahap ini dilakukan untuk menghapus simbol seperti strip "-", dalam kurung "(" dan luar kurung ")", sehingga kata-kata tersebut tidak menyambung ketika dilakukan tokenisasi nantinya. Contoh menghapus ditunjukkan pada Tabel 3.2.

Tabel 3.2 Contoh menghapus simbol

Sebelum	**Sesudah**
1 kaleng sarden besar (saya: merkmakarel warna merah) --2 ikat sawi putih--Bumbu Halus: --8 siung bawang merah--3 siung bawang putih--9 buah cabe rawit--4 buat cabe merah besar--Secukupnya gula, garam--Secukupnya minyak goreng--	1 kaleng sarden besar saya merk makarel warna merah 2 ikat sawi putih Bumbu Halus 8 siung bawang merah 3 siung bawang putih 9 buah cabe rawit 4 buat cabe merah besar Secukupnya gula garam Secukupnya minyak goreng

c. Menghapus spasi berlebih

Tahap ini digunakan untuk menghilangkan spasi berlebih pada data. Ketika terdapat spasi berlebih dari satu seperti "cabe merah besar", maka kelebihan spasi tersebut akan dihapus sehingga hanya menjadi satu spasi dan dapat mengurangi dimensi data. Contoh menghilangkanspasi berlebih ditunjukkan pada Tabel 3.3.

Tabel 3.3 Contoh menghapus spasi berlebih

Sebelum	Sesudah
Bahan untuk biang 400 ml air kaldu udang rebus air dengan secukupnya udangsegar 200 gede rasa terigu Secukupnya garam gula Kaldu bubuk boleh skip Bahan adonan pempek 3 btr telur kocok lepas 5 siung bawang putih haluskan 400 gede rasa tepung tapioka	Bahan untuk biang 400 ml air kaldu udang rebus air dengan secukupnya udang segar 200 gede rasa terigu Secukupnya garam gula Kaldu bubuk boleh skip Bahanadonan pempek 3 btr telur kocok lepas 5 siung bawang putih haluskan 400 gede rasa tepung tapioka

d. Tokenisasi

Tokenisasi merupakan proses untuk membagi teks atau kata menjadi beberapa bagian atau unit yang lebih kecil. Beberapa teks yang telah dibagi menjadi beberapa unit kecil tadi disebut sebagai token. Contoh tokenisasi ditunjukkan pada Tabel 3.4.

Tabel 3.4 Contoh tokenisasi

Sebelum	Sesudah
1 4 kg ayam 3 buah cabai hijau besar 7 buah cabai merah rawit 3 siung bawang putih 2 siung bawang merah secukupnya Gula secukupnya Garam 1 4 buah tomat merah secukupnya Air secukupnya Minyak goreng	[‘1’, ‘4’, ‘kg’, ‘ayam’, ‘3’, ‘buah’, ‘cabai’, ‘hijau’, ‘besar’, ‘7’, ‘buah’, ‘cabai’, ‘merah’, ‘rawit’, ‘3’, ‘siung’, ‘bawang’, ‘putih’, ‘2’, ‘siung’, ‘bawang’, ‘merah’, ‘secukupnya’, ‘Gula’, ‘secukupnya’, ‘Garam’, ‘1’, ‘4’, ‘buah’, ‘tomat’, ‘merah’, ‘suckupnya’, ‘Air’,‘secukupnya’, ‘Minyak’, ‘goreng’]

3.2.3 Pelabelan Entitas

Setelah melakukan *preprocessing,* tahap selanjutnya yaitu melakukan pelabelan entitas pada data. Tahap ini akan memberikan label entitas pada data yang telah melalui proses tokenisasi sesuai dengan kategori entitasnya. Pelabelan entitas dilakukan secara manual. Kategori entitas pada penelitian ini dibagi menjadi tiga entitas, yaitu bahan masakan, bumbu masakan dan alat. Berikut ini merupakan penjelasan dari kategori entitas tersebut:

a. Bahan makanan

Kategori ini merupakan kategori yang berisi mengenai bahan-bahan masakan seperti ayam, ikan, brokoli, tahu, tempe, telur, tepung dan minyak goreng.

b. Bumbu masakan

Entitas ini merupakan kategori yang berisi mengenai bumbu-bumbu masakan seperti cengkeh, daun salam, merica, oregano, garam dan paprika.

c. Alat

Dalam kategori alat ini akan berisi nama-nama alat. Alat dapat berupa alat yang digunakan dalam memasak seperti semdok makan, gelas dan mangkok.

Dalam melakukan pelabelan entitas ini, data dilabeli menggunakan format *Beginning Inside Outside* (BIO) karena entitas dapat memiliki dari beberapa token (Hoesen & Purwarianti, 2019). Format BIO merupakan cara melabeli dengan menggolongkan entitas berdasarkan urutan posisi kata dalam kalimat. Apabila kata entitas tersebut berada di urutan pertama maka akan di beri label "B" yang berarti *Begin*, sedangkan jika kata entitas lebih dari satu dan urutannya terletak setelah kata entitas pertama, yaitu di urutan kedua dan seterusnya maka akan diberi label "I" yang berarti *Inside* dan jika kata entitas bukan termasuk yang ada pada kategori maka akan diberi label "O" yang berarti *Other*. Tabel 3.5 merupakan contoh format BIO dan label yang akan digunakan dalam penelitian ini.

Tabel 3.5 Jenis macam label dan penjelasan

Label	**Keterangan**	**Contoh**
B-BAH	Kata pertama (*Beginning*) pada entitas bahan masakan diberi label B-BAH	Ayam Broiler: Ayam: B-BAH Broiler: I-BAH
I-BAH	Kata kedua dan seterusnya (*Inside*) pada entitas bahan masakan diberi label I-BAH	

B-BUM	Kata pertama (*Beginning*) pada entitas bumbu masakan diberi label B-BUM	Daun Bawang: Daun: B-BUM Bawang: I-BUM
I-BUM	Kata kedua dan seterusnya (*Inside*) pada bumbu masakan diberi label I-BUM	
B-ALT	Kata pertama (*Beginning*) pada entitas alat diberi label B-ALT	Sendok Makan: Sendok: B-ALT Makan: I-ALT
I-ALT	Kata kedua dan seterusnya (*Inside*) pada entitas alat diberi label I-ALT	
O	Kata-kata yang tidak termasuk dalam label entitas nama tempat wisata, nama lokasi, fasilitas dan suasana akan diberi label O (*Outside*)	1 lembar 1: O Lembar: O

3.2.4 Ekstraksi Fitur

Ekstraksi fitur yang digunakan dalam penelitian ini adalah *Word2Vec*. Fitur *Word2Vec* ini berguna untuk mengubah data yang tadinya teks menjadi sebuah vektor. Tabel 3.6 menunjukkan contoh *Word2Vec* dalam beberapa kata dengan vektor berdimensi 5. Setiap dimensi mewakili sebuah makna, sedangkan bobot numerik kata pada dimensi tersebut akan menangkap kedekatan keterkaitan hubungannya dengan makna yang diwakilkan. Akan tetapi, makna apa yang diwakilkan pada setiap dimensinya tidak dapat terdefinisi secara jelas oleh manusia.

Tabel 3.6 Contoh *Word2Vec*

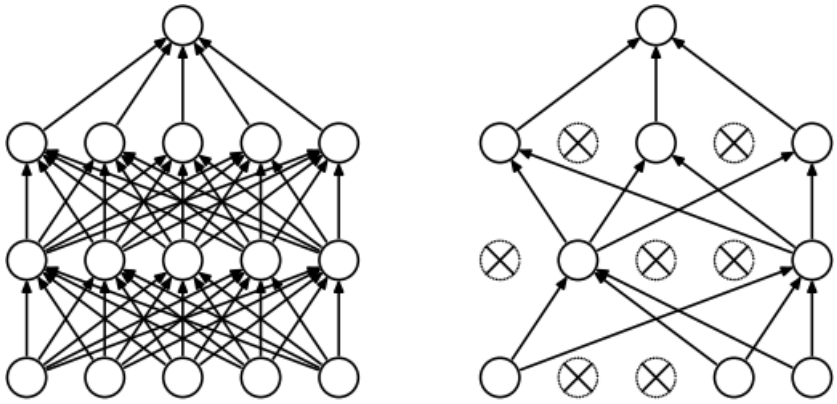
Kata	Vektor
Garam	[-0.27508005, 0.7812888, -0.023236627, -0.5496242, 1.055]
Cabe	[-0.011669971, 0.73168933, 0.67134243, -0.0016824865, 1.20435]
Ikan	[0.27758798, -1.0282282, 0.4605686 0.42941102, -1.0479026]
Sendok	[-0.99666685, -0.26300707, -0.1798355, -0.6347489, 1.8852518]
Sapi	[0.034535985, -0.25741088, 0.21693623, 0.11281757, -0.23945619]
Tuna	[0.08464525, -0.35611892, 0.058538403, 0.09098392, -0.41207004]
Paprika	[0.008741743, -0.049658895, 0.13670827, 0.082177505, -0.07861546]

3.2.5 Pemodelan Named Entitty Recognition

Dalam melakukan pemodelan, sebelum membangun model maka langkah pertama yang dilakukan adalah *splitting data. Splitting data* dilakukan untuk membagi data menjadi dua yaitu data latih dan data uji. Kemudian data latih akan dibagi lagi sebagaian menjadi data *validation.* Data *validation* merupakan data untuk melakukan validasi hasil prediksi pada model saat melakukan proses pengujian. Data latih yang lainnya yang tidak termasuk di data *validation* akan digunakan ketika melatih model. Setelah model terlatih oleh data latih dan data validasi akan dilakukan pengujian pada model menggunakan data uji. Pengujian ini dilakukan untuk mengetahui performa model yang telah dibuat sebelumnya.

Setelah melakukan splitting data langkah selanjutnya adalah membangun model. Dalam penelitian ini model dibangun meggunakan dua metode yaitu Bidirectional Long Short- Term Memory (BiLSTM) dan Bidirectional Encoder Representations from Transformers (BERT). Pada metode BiLSTM akan digunakan beberapa lapisan, yaitu lapisan embedding, lapisan BiLSTM dan lapisan dense. Model ini akan menggunakn fitur Word2Vec, dengan Word2Vec ini data yang berupa teks akan diubah menjadi vektor. Data yang telah menjadi vektor akan dilatih dan menghasilkan sebuah kamus. Nantinya kata-kata yang memiliki konteks yang mirip

akan memiliki jumlah vektor yang berdekatan karena dalam Word2Vec, kata-kata yang memiliki kaitan yang sama, vektornya akan berdekatan. Kemudian kamus tadi yang berisi vektor akan dijadikan sebagai bobot dalam lapisan embedding. Lapisan embedding akan memetakan setiap kata menjadi bentuk vektor dimana vektor tersebut akan mempresentasikan kata sesuai dengan yang ada di dalam kamus Word2Vec tadi. Setelah vektor-vektor tadi melalui lapisan *embedding*, vektor-vektor akan masuk ke lapisan BiLSTM.Lapisan BiLSTM ini terdiri dari dua lapisan LSTM yaitu lapisan *forward* dan lapisan *backward*. Lapisan *forward* adalah lapisan yang akan bergerak maju dari awal kalimat ke akhir kalimat sehingga menangkap informasi dari masa lalu, sedangkan lapisan *backward* akan bergerak mundur dari akhir kalimat ke awal kalimat sehingga menangkap informasi dari masadepan. Kemudian keluaran dari lapisan BiLSTM akan diteruskan ke lapisan *dense*. Lapisan *dense* merupakan lapisan keluaran yang nantinya akan menghubungkan semua keluaran dari lapisan sebelumnya ke semua *node* pada lapisan *dense*. Untuk mencegah terjadi *overfitting* pada model, digunakan teknik *dropout. Overfitting* merupakan keadaan dimana ketika model mempelajari data latih dan mendapatkan hasil akurasi yang sangat tinggi, tetapi ketika model memproses sebuah data yang baru, akurasi yang dihasilkan menjadi rendah. Teknik *dropout* mampu mencegah *overfitting* dan dapat membantu model lebih efisien. *Dropout* mengatasi *overfitting* dengan *neuron* yang terpilih secara acak dihilangkan sementara dari jaringan selama proses pelatihan(Srivastava et al., 2014). Gambar 3.3 menunjukkan bagaimana perbedaan ketika menggunakan *dropout* dan tidak menggunakan *dropout.*



Gambar 3.3 Teknik dropout

Sumber: (Srivastava et al., 2014)

Pada metode BERT, pemodelan dilakukan dengan *import* transformer yang menyediakan *package* kelas BertForTokenClassification untuk NER. BertForTokenClassification merupakan *fine-tuning* model yang mengelilingi BertModel dan menambahkan *token-level classifier* ke dalam BertModel. *Token-level classifier* merupakan lapisan yang mengambil input dari *hidden state* terakhir dari urutan. Untuk mencegah *overfitting* pada model, model BERT akan menggunakan teknik *dropout*. Dalam penelitian ini, akan terdapat beberapa skenario yang dibuat. Skenario bertujuan untuk mengetahui serta mendapatkan model dengan performa terbaik dalam melakukan NER. Skenario awal pada model BiLSTM akan menggunakan *Word2Vec* berupa *skip-gram* karena akurasi *skip-gram* memiliki akurasi lebih tinggi daripada CBOW, walaupun secara teori CBOW lebih diunggulkan karena dapat mencakup lebih banyak kata yang bener (Seok et al., 2016). Jumlah unit BiLSTM yang digunakan sebanyak 128 unit seperti yang digunakan pada penelitian(Wang et al., 2019). Nilai *dropout* yang digunakan pada lapisan LSTM sebesar 0,5 untuk mencegah *overfitting*. Jumlah lapisan *dense* sebanyak 7, sesuai dengan banyaknya jenis label pada penelitian ini. Fungsi aktivasi yang digunakan adalah *softmax*, ini dipilih karena fungsi *softmax* memiliki kinerja yang lebih baik dari fungsi aktivasi lainnya (Gunawan et al., 2018). Fungsi optimasi yang digunakan adalah *Adam*, ini dipilih karena fungsi optimasi ini memiliki performa yang lebih baik dari fungsi optimasi lainnya (Reimers & Gurevych, 2017). Cacah *epoch* yang digunakan adalah 30 dan *batch size* yang digunakan adalah 32. Skenario awal pada model BERT akan menggunakan *dropout* sebesar 0,5 untuk mecegah *overfitting*. Fungsi optimasi yang digunakan adalah *AdamW*. Cacah *epoch* yang digunakan adalah 4 dan *batch size* sebesar 32 seperti yang dilakukan oleh penelitian (Devlin et al., 2019). Kemudian akan terdapat 6 skenario

yang dilakukan dalam penelitian ini, model BiLSTM akan ada 3 skenario dan model BERT akan ada 3 skenario. Skenario ini membandingkan performa model dari perbedaan algoritma dan parameter yang digunakan. Skenario di model BiLSTM merupakan penggunaan jumlah unit, fungsi optimasi dan *learning rate* sedangkan untuk model BERT penggunan jumlah *epoch*, jumlah *batch* dan *dropout*. Skenario 1 model BiLSTM akan melakukan perbandingan pada jumlah unit LSTM yang digunakan, yaitu sebanyak 100 dan 120. Skenario 2 model BiLSTM akan membandingkan *learning rate* yang digunakan pada model, *learning rate* yang akan dibandingkan ada tiga, yaitu *learning rate* dengan nilai 0,01, 0,001 dan 0,0001. *Learning rate* merupakan parameter yang berguna untuk mengatur seberapa cepat model dalam belajar. *Learning rate* sangat mempengaruhi kecepatan pembelajaran pada model, jika *learning rate* kecil akan mendapatkan hasil yang optimal tetapi membutuhkan waktu yang lama untuk model belajar, sedangkan jika *learning rate* besar dapat mempercepat model dalam belajar tetapi mendapatkan hasil kurang optimal (Saputra, 2020). Skenario 3 model BiLSTM akan membandingan fungsi optimasi yang digunakan, yaitu *Adam* dan *Nadam*. Pada skenario 1 model BERT akan melakukan perbandingan jumlah *epoch* yang digunakan, yaitu 4, 10 dan 20. Skenario 2 model BERT akan melakukan perbandingan jumlah *batch size* yang digunakan, yaitu 32 dan 64. Skenario 3 model BERT akan membandingkan jumlah *dropout* yang digunakan untuk mengetahui nilai *dropout* yang cocok. Konfigurasi *hyperparameter* yang memiliki performa terbaik dalam setiap skenario akan digunakan dalam skenraio-skenario berikutnya. Skema skenario dari model BiLSTM terdapat pada Tabel 3.7 dan skema skenario dari model BERT terdapat pada Tabel 3.8.

Tabel 3.7 Rangkuman scenario model BiLSTM

Skenario	***Hyperparameters***	**Konfigurasi**
1	Unit LSTM	100
		128
2	*Learning rate*	0,01
		0,001
		0,0001
2	Fungsi optimasi	*Adam*
		Nadam

Tabel 3.8 Rangkuman scenario model BERT

Skenario	***Hyperparameters***	**Konfigurasi**
1	*Epoch*	4
		10
		20
2	*Batch size*	32
		64
3	*Dropout*	-
		0,25
		0,5

3.2.6 Evaluasi Model

Untuk mengetahui seberapa baik kinerja model NER maka akan diukur dengan menggunakan F1-*Score*. Pengukuran F1-*Score* dipilih karena menurut penelitian (Seok et al., 2016) penggunaan F1-*Score* lebich cocok digunakan daripada akurasi, hal ini karena sebagian besar label pada NER merupakan label O yang merupakan bukan entitas bernama sehingga berhasilmendapatkan akurasi yang tinggi. F1-*Score* diperoleh dengan menghitung nilai rata-rata dari *precision* dan *recall*. *Precision* merupakan banyaknya hasil positif yang bener dibagi dengan banyaknya seluruh hasil positif. *Recall* adalah banyaknya hasil data positif yang bener dibagi dengan banyaknya hasil semua data yang seharusnya bernilai positif. Perhitungan F1-*Score* terdapat pada persamaan. Karena keluaran NER terdiri dari beberapa kelas label, maka untuk mencari F1-*Score* dibutuhkan parameter *average* yang meliputi *micro, macro dan weighted*. *Micro* akan menghitung F1-*Score* dengan mempertimbangkan total *true positive, false negative* dan *false positive*. *Macro* akan menghitung F1-*Score* untuk setiap labelnya dan mengembalikan rata-rata tanpa mempertimbangkan proporsi untuk setiap label dalam *dataset*. *Weighted* akan menghitung F1-*Score* untuk setiap label dan mengembalikan rata-rata dengan mempertimbangkan rata-rata dengan proporsi untuk setiap label.

$$F1 - Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (3.1)$$

3.2.7 Deteksi Entitas

Dalam tahapan ini, akan melakukan deteksi entitas yang terdapat pada data berupa bahan masakan, bumbu masakan dan alat dalam suatu masukan. Tahapan ini akan menggunakan model dengan nilai F1-*Score* terbaik yang telah dibuat pada skenario sebelumnya.

BAB IV
HASIL DAN PEMBAHASAN

4.1 Dataset

Data pada penelitian ini menggunakan dataset yang diambil dari *website* Kaggle. Dataset yang diambil dari Kaggle merupakan data yang berupa resep masakan Indonesia yang dikategorikandalam tujuh kategori. Data yang digunakan dalam penelitian ini berjumlah 3999 resep masakan Indonesia. Resep masakan tersebut ditokenisasi menjadi penggalan kata dan menghasilkan sebanyak 196.696 data yang berupa kata-kata yang kemudian dibagi menjadi dua data, yaitu data latih dan data uji. Data latih dan data uji dibagi sebanyak 0.2, yang berarti data latih sebanyak 80% dan data uji sebanyak 20% sehingga sampel data latih berjumlah 3199 dan data uji berjumlah 199.

4.2 Preprocessing

Setelah selesai mengumpulkan data, langkah berikutnya adalah melakukan *preprocessing. Preprocessing* yang dilakukan yaitu:

a. Menghapus karakter NON-ASCII

Tahap pertama dalam *preprocessing* adalah menghapus karakter NON-ASCII. Tahap ini digunakan untuk menghapus karakter-karakter NON-ASCII. Gambar 4.1 menunjukkan implementasi kode untuk menghapus karakter NON-ASCII.

```
def remove_non_ascii(str): # Berguna untuk menghapus code ascii
return "".join(i for i in str if ord(i)<128
```

Gambar 4.1 Kode program menghapus non-ascii

b. Menghapus simbol

Tahap ini digunakan untuk menghapus simbol yang terdapat pada data. Tahap ini dilakukan untuk menghapus simbol seperti strip “-”, dalam kurung “(” dan luar kurung “)”. Gambar 4.2 menunjukkan implementasi kode untuk menghapus simbol.

```
def remove_symbol(str):
str = re.sub('[^\/w]', ' ', str)
return str;
```

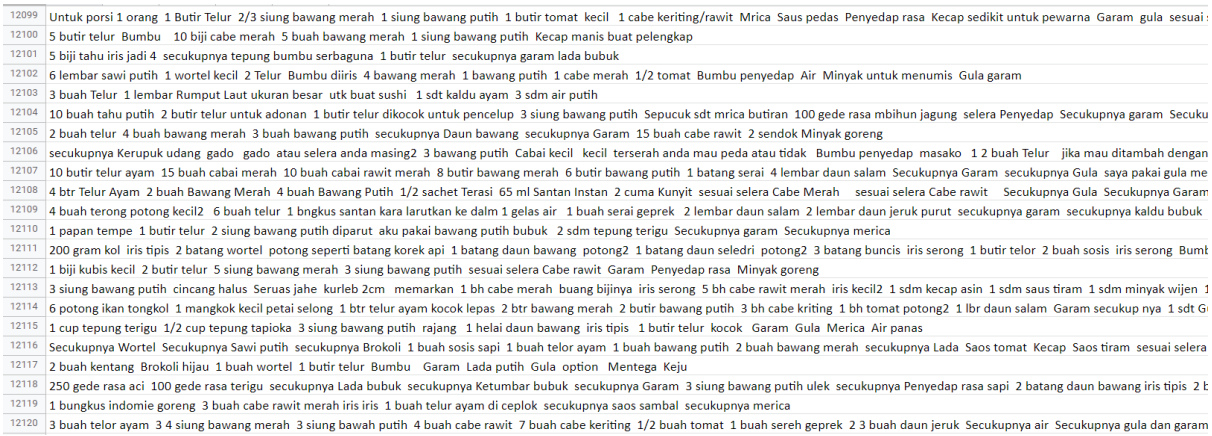
Gambar 4.2 Kode program untuk menghapus symbol

c. Menghapus spasi berlebih

Tahap ini digunakan untuk menghilangkan spasi berlebih pada data. Gambar 4.3 menunjukkan kode untuk menghilangkan spasi berlebih.

```
def remove_add_space(str): # Berguna untuk menghapus spasi berlebih
str = re.sub('[\s]+', ' ', str)
return str;
return 0;
```

Gambar 4.3 Kode program menghapus spasi berlebih



Gambar 4.4 Hasil *preprocessing*

d. Tokenisasi

Langkah terakhir dalam tahap *preprocessing* adalab tokenisasi. Data yang berupa kumpulan kalimat akan dipecah menjadi kata-kata yang kemudian disimpan dalam file .csv serta nomor kalimat yang menandakan urutan kalimat tersebut. Gambar 4.5 menunjukkan kode program tokenisasi.

```
ct = CRFTagger() ct.set_model_file('all_indo_man_tag_corpus_model.crf.tagger')
def tagging(sent):
xx = word_tokenize(sent)
hasil = ct.tag_sents([xx])
return hasil
```

Gambar 4.5 Kode program tokenisasi

Sentence 3497	2
	papan
	tempe
	ukuran
	besar
	6
	siung
	bawang
	merah
	4
	siung
	bawang
	putih
	1
	batang
	sereh
	1
	buah
	daun
	jeruk
	1

Gambar 4.6 Hasil tokenisasi

4.3 Pelabelan Entitas

Setelah selesai melalui tahap *preprocessing*, tahap selanjutnya yaitu melakukan pelabelan pada data dengan memberi label sesuai dengan kategori entitas masing-masing. Label yang akan diberikan pada tiap kata ada 7, yaitu B-BAH, B-BUM, B-ALT, I-BAH, I-BUM, I-ALT dan O. Gambar 4.6 menunjukkan data yang sudah ditokenisasi sesuai dengan kategori entitas masing-masing. Sedangkan pada Tabel 4.1 menunjukkan kategori label dan jumlah banyaknya label per kategori. Dapat dilihat pada Tabel 4.1 jumlah entitas Alat lebih sedikit dibandingkan entitas lain, hal ini dikarenakan enitas Alat di data hanya ada dua atau tiga sehingga entitas Bahan dan Bumbu lebih banyak daripada entitas Alat.

Sentence 3995	4	CD	O
	bungkus	NN	O
	Mie	NNP	B-BAH
	telor	NN	I-BAH
	8	CD	O
	buah	NND	O
	telur	NN	B-BAH
	rebus	VB	O
	Secukupnya	NN	O
	bakso	FW	B-BAH
	udang	FW	I-BAH
	sy	FW	O
	60	CD	O
	pcs	NN	O
	1	CD	O
	ikat	NN	O
	besar	JJ	O
	sawi	VB	B-BAH
	hijau	VB	I-BAH
	250	CD	O
	gram	NN	O
	wortel	NN	B-BAH
	iris	NN	O
	korek	NN	O
	api	NN	O
	Kuah	NNP	O
	Tomyum	NNP	O
	15	CD	O
	siung	NN	O
	bawang	NN	B-BUM
	putih	VB	I-BUM

Gambar 4.7 Hasil pelabelan entitas

Tabel 4.1 Jumlah token pada setiap entitas

Label	Jumlah Token
B-BAH	16955
I-BAH	9558
B-BUM	32884
I-BUM	20484
B-ALT	917
I-ALT	341
O	115557

4.4 Mempersiapkan Data Masukan

Dalam membangun model NER, dilakukan persiapan untuk data masukan terlebih dahulu. Terdapat tiga proses dalam mempersiapkan data masukan, yaitu masukan pada model BiLSTM, masukan pada *Word2Vec* dan masukan pada model BERT. Berikut merupakan penjelasan prosesnya:

Data masukan pada model BiLSTM

Data yang akan digunakan sebagai masukan pada model akan dipersiapkan terlebih dahulu akan dibentuk menjadi sebuah *list*. Dalam *list* tersebut berisi *tuple* yang menggunakan

kelas bernama SentenceGetter. Kelas ini berguna untuk memisahkan antara tiap kalimat satu dari kalimat lainnya, kalimat di dalam *list* nantinya akan disebut sebagai sampel. Banyaknya jumlah sampel yang ada di penelitian ini yaitu total sebanyak 3999 sampel. Kemudian setiap kata-kata dan label yang terdapat di dalam kalimat akan diubah menjadi indeks angka, dimana angka tersebut mewakili setiap kata dan label. Untuk memastikan bahwa setiap sampel memiliki panjang yang sama, akan digunakan fungsi *pad_sequences ()*. Panjang sampel yang digunakan dalam BiLSTM adalah 120. Fungsi *pad_sequences ()* akan memotong sampel yang memiliki panjang dari 120, sedangkan apabila ada sampel yang memiliki panjang dari 120 maka akan ditambah menggunakan angka 0. Gambar 4.8 Kode program untuk persiapan masuk data ke model BilSTM menunjukkan kode program dalam tahapan ini.

```
class SentenceGetter(object):
       def_init_(self, data):
           self.n_sent = 1
           self.data = data
           self.empty = False
           agg_func = lambda s: [(w, t) for w, t in zip(s["Word"].values.tolist(),
                                                        s["Tag"].values.tolist())]
           self.grouped = self.data.groupby("Sentence #").apply(agg_func)
           self.sentences = [s for s in self.grouped]

           def get_next(self):
               try:
                   s = self.grouped["Sentence {}".format(self.n_sent)]
                   self.n_sent += 1
                   return s
               except:
                   return None

getter = SentenceGetter(data)
sentences = getter.sentences

words = sorted(set(WORD))
words.append("ENDPAD")
n_words = len(words); n_words
tags = sorted(set(TAG))
n_tags = len(tags); n_tags

max_len = 120
word2idx = {w: i + 1 for i, w in enumerate(words)}
tag2idx = {t: i for i, t in enumerate(tags)}

X_one = [[word2idx[w[0]] for w in s] for s in sentences]
X = pad_sequences(maxlen=max_len, sequences=X_one, padding="post", value=n_words-1)

y_one = [[tag2idx[w[2]] for w in s] for s in sentences]
y_two     =     pad_sequences(maxlen=max_len,     sequences=y_one,     padding="post",
value=tag2idx["O"])
```

Gambar 4.8 Kode program untuk persiapan masuk data ke model BilSTM

Data masukan kelas *Word2Vec*

Kemudian data masukan pada kelas *Word2Vec* akan dipersiapkan terlebih dahulu dengan dibentuk menjadi sebuah *list* yang menjadi sebuah sampel menggunakan kelas *SentenceGetter2* yang bertujuan untuk memisahkan antara setiap kalimat satu dengan kalimat lainnya, sama seperti yang dilakukan pada langkah sebelumnya. Gambar 4.9 Kode program untuk persiapan data masuk ke kelas *Word2Vec* merupakan kodeprogram dalam tahapan ini.

```
class SentenceGetter2(object):

    def_init_(self, data):
        self.n_sent = 1
        self.data = data
        self.empty = False
        agg_func = lambda s: [w for w in s["Word"].values.tolist()]
        self.grouped = self.data.groupby("Sentence #").apply(agg_func)
        self.sentences = [s for s in self.grouped]


    def get_next2(self):
        try:
            s = self.grouped["Sentence {}".format(self.n_sent)]
            self.n_sent += 1
            return s
        except:
            return None


getter2 = SentenceGetter2(data)
sentences2 = getter2.sentences
```

Gambar 4.9 Kode program untuk persiapan data masuk ke kelas *Word2Vec*

Data masukan pada model BERT

Kemudian data masukan pada model BERT akan dipersiapkan dengan dibentuk menjadi sebuah list yang menjadi sebuah sampel menggunakan kelas SentenceGetter. yang bertujuan untuk memisahkan antara setiap kalimat dari kalimat lainnya, sama seperti langkah sebelumnya.

Gambar 4.10 menunjukkan kode program dalam tahapan ini.

```
class SentenceGetter(object):

    def_init_(self, data):
    self.n_sent = 1
    self.data = data
    self.empty = False
    agg_func   =   lambda s:[(w,p,t) for w, p, t in

    zip(s["Word"].values.tolist(),

    s["POS"].values.tolist(),

    s["Tag"].values.tolist())]
            self.grouped     =     self.data.groupby("Sentence     #").apply(agg_func)
            self.sentences = [s for s in self.grouped]

def get_next(self):try:
s = self.grouped["Sentence: {}".format(self.n_sent)]self.n_sent += 1
return sexcept:
return None
```

Gambar 4.10 Kode program persiapan data masuk ke model BERT

4.5 Ekstraksi Fitur

Setelah data sudah dipersiapkan, tahap selnajutnya adalah melakukan ekstraksi fitur. Ektraksi fitur yang digunakan adalah teknik *Word2Vec*. Tujuannya adalah untuk mengubah setiap kata menjadi vektor, dimana vektor tersebut merepresentasikan kata-kata tersebut. Dalam melakukan teknik *Word2Vec*, digunakan *library Gensim* dengan *Word2Vec* sebagai kelasnya. Panjang vektor yang digunakan yaitu sebanyak 100 yang merupakan ukuran *default* dari kelas tersebut. Gambar 4.11 merupakan kode program untuk melakukan tahap ini.

```
model_w2v = gensim.models.Word2Vec(sentences=sentences2, size=100, window=10,
                                   workers=1, min_count=1, sg=0, seed=1)
```

Gambar 4.11 Kode program *Word2Vec*

4.6 Pemodelan Named Entity Recognition

Dalam melakukan tahap pemodelan NER, ada beberapa tahap yang dilakukan. Berikut merupakan tahapan yang dilakukan dalam pemodelan NER:

a. Splitting Data

Untuk membangun sebuah model, maka data akan dibagi menjadi dua bagian, yaitu data latih dan data uji. Pada model BiLSTM dan BERT, pembagian data menggunakan *library Sckit- Learn* dengan nama kelas *train_test_split*. Dengan kelas ini, data dapat dibagi menjadi data latih dan data uji secara acak dalam proporsi. Dalam kasus ini data latih dan data uji dibagi sebanyak 0.2, yang berarti data latih sebanyak 80% dan data uji sebanyak 20% sehingga sampel

data latih berjumlah 3199 dan data uji berjumlah 199. Gambar 4.12 dan Gambar 4.13 menunjukkan kode program *data splitting* pada model BiLSTM dan BERT.

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state = 1)
```

Gambar 4.12 Kode program *splitting data* model BiLSTM

```
from sklearn.model_selection import train_test_split

tr_inputs, val_inputs, tr_tags, val_tags = train_test_split(input_ids, tags,
random_state=2018, test_size=0.2)
```

Gambar 4.13 Kode program *splitting data* model BERT

b. Membangun Model BiLSTM

Dalam membangun model BiLSTM, digunakan tipe *Sequential()*. Tipe tersebut akan membangun model dari lapisan ke lapisan secara berurutan. Untuk menambahkan lapisan baru pada model, akan digunakan fungsi *add()*. Activation merupakan fungsi aktivasi pada model yang berguna untuk mengetahui apakah jaringan tertentu diaktifkan atau tidak. Fungsi aktivasi akan melakukan transformasi *non linear* dari data masukan sehingga dapat kemungkinan bahwa jaringan akan belajar lebih baik. Selanjutnya lapisan *dense* yang merupakan lapisan keluaran yang berfungsi untuk menghubungkan semua *node* lapisan sebelumnya ke *node* lapisan ini. Setelah selesai konfigurasi lapisan, langkah selanjutnya adalah *compile* model. Dalam compile model ini akan terdapat beberapa parameter yang digunakan, yaitu *optimizer*, *loss* dan *metrics. Optimizer* atau fungsi optimasi digunakan untuk mengganti attribut neural network seperti learning rate serta dapat membantu menurunkan nilai *loss*. *Loss* digunakan untuk menghitung estimasi error pada *neural network* yang digunakan dalam perhitungan gradien untuk memperbarui bobot *neural network*. *Metrics* digunakan untuk menilai performa model. Dalam model ini *optimizer* yang digunakan adalah *Adam*, *Loss* yang digunakan adalah *categorical_cassetropy* dan Metrics yang digunakan berupa *accuracy*. Gambar 4.14 menunjukkan kode program membangun model BiLSTM.

```
model = Sequential()
model.add(embedding_layer)
model.add(Bidirectional(LSTM(128, return_sequences=True)))
model.add(Dropout(0.5))
model.add(Dense(n_tags, activation='softmax'))
```

```
model.compile(optimizer='adam', loss='categorical_crossentropy',
metrics=['accuracy'])
model.summary()
```

Gambar 4.14 Kode program membangun model BiLSTM

c. Membangun model BERT

Dalam membangun model BERT, dilakukan dengan menggunakan *library* dari *transformers* dengan nama kelas *BertForTokenClassification*. Kelas ini digunakan khusus untuk melakukan NER dan merupakan *fine-tuning* model yang mengelilingi *BertModel* dan menambahkan *token-level classifier*. *Token-level classifier* merupakan suatu lapisan yang mengambil input dari *hidden state* urutan terakhir. Selanjutnya tahap berikut untuk membangun model BERT adalah *load bert_base_case* model yang sudah di *pre-train* di dalam *library transformers*. Fungsi optimasi yang digunakan pada model ini adalah fungsi *Adam*. Gambar 4.15 menunjukkan kode program membangun model BERT

```
import transformers
from transformers import BertForTokenClassification, AdamW, TFBertModel


model = BertForTokenClassification.from_pretrained(
        'bert-base-case',
            num_labels = len(tag2idx),
            output_attentions = False,
            output_hidden_states = False,
            attention_probs_dropout_prob = 0.2,
            hidden_dropout_prob = 0.2
)


model.cuda();
```

Gambar 4.15 Kode program membangun model BERT

d. Melatih Model

Dalam melatih model BiLSTM, akan digunakan fungsi *fits ()* dengan 5 parameter, yaitu *X_train*, *y_train*, *batch size*, *epoch* dan *validation_split*. *X_train* merupakan data masukan sedangkan *y_train* merupakan target data. Parameter *validation_split* berguna untuk membagi antara data latih dan data validasi. Data latih dan data uji dibagi dengan ukuran 0.2 dimana 80% sebagai data latih dan 20% sebagai data validasi. *Epoch* akan menentukan berapa kali data latih akan dilatih oleh model. Akan tetapi setiap satu kali epoch akan terlalu besar jika diproses secara bersamaan, oleh karena itu harus dibagi menjadi beberapa bagian *batch* menggunakan *batch size*. *Batch size* digunakan untuk membagi data menjadi beberapa *batch* yang lebih kecil

ketika melakukan pelatihan. Gambar 4.16 menunjukkan kode program untuk tahap melatih model BiLSTM.

```
history = model.fit(X_train, np.array(y_train), batch_size=32, epochs=30,
validation_split=0.2, verbose=1)
```

Gambar 4.16 Kode program melatih model BiLSTM

Dalam melatih model BERT, ada beberapa fungsi yang digunakan. Pertama membuat model dalam *mode train* dengan menggunakan *model.train()*. Selanjutnya menambahkan *batch*ke GPU supaya dapat meningkatkan kinerja model dengan menggunakan *batch = tuple(t.to(device) for t in batch)*. Kemudian sebelum melaksanakan *backward pass* model akanmenghapus semua gradien yang sudah dihitung sebelumnya dengan fungsi *model.zero_grad()*.Setelah sudah menghapus perhitungan gradien sebelumnya, langkah berikutnya adalahmelakukan *forward pass* untuk mendapatkan nilai *loss*. Langkah selanjutnya untukmendapatkan nilai *loss* digunakan fungsi *loss=outputs()*. Setelah mendapatkan nilai *loss*,langkah selanjutnya adalah melakukan *backward pass* untuk menghitung gradien. Kemudian untuk mencegah masalah '*exploding gradient*' dalam model digunakan fungsi *torch.nn.utils.clip_grad_norm_(parameters=model.parameters(),max_norm=max_grad_norm)*. Kemudian akan digunakan fungsi *optimizer.step()*. Tujuan fungsi *optimizer.step()* adalah membuat *optimizer* melakukan pengulangan ke semua parameters dan melakukan update pada semua parameters. Selanjutnya dilakukan *update* pada *learning rate* untuk menyesuaikan berdasarkan beberapa *epochs*. Gambar 4.17 menunjukkan kode program untuk tahap melatih model BERT.

```
from seqeval.metrics import f1_score, accuracy_score
## Menyimpan loss value dan validasi loss value
loss_values, validation_loss_values = [], []

for _ in trange(epochs, desc="Epoch"):

    # Masukkan model ke mode pelatihan.model.train()
    # Reset the total loss for this epoch.total_loss = 0

    # Training loop
    for step, batch in enumerate(train_dataloader):
        # menambahkan batch ke gpu
        batch = tuple(t.to(device) for t in batch)
        b_input_ids, b_input_mask, b_labels = batch
        # Selalu bersihkan gradien yang dihitung sebelumnya sebelum melakukan passing
mundur.
        model.zero_grad()
```

```
        # forward pass
        # This will return the loss (rather than the model output)
        # because we have provided the `labels`.
        outputs = model(b_input_ids, token_type_ids=None,
                        attention_mask=b_input_mask, labels=b_labels)
        # get the loss
        loss = outputs[0]
        # Perform a backward pass to calculate the gradients.
        loss.backward()
        #track train loss
        total_loss += loss.item()
        # Clip the norm of the gradient
        #This   is   to   help   prevent   the   "exploding   gradients"   problem.
        torch.nn.utils.clip_grad_norm_(parameters=model.parameters(),
Max_norm=max_grad_norm)
        # update parameters
        optimizer.step()
        # Update the learning rate.
        scheduler.step()

    # Calculate the average loss over the training data.
    avg_train_loss = total_loss / len(train_dataloader)
    print("Average train loss: {}".format(avg_train_loss))

    #    Store    the    loss    value    for    plotting    the    learning    curve.
    loss_values.append(avg_train_loss)
```

Gambar 4.17 Kode program melatih model BERT

4.7 Evaluasi Model

Tahap evaluasi dilakukan untuk mengukur kinerja dari model BiLSTM dan BERT yang sudah dibangun. Performa model dari setiap skenario yang sudah dibuat akan dilakukan perbandingan menggunakan rata-rata F1-*Score* yang telah diperoleh, hal ini bertujuan untuk mendapatkan model terbaik dalam melakaukan NER.

Model BiLSTM

a. Skenario 1

Pada skenario ini dilakukan perbandingan jumlah unit LSTM. Hasil evaluasi F1-*Score* dapat dilihat di Tabel 4.2. Dapat diketahui dari tabel tersebut, model menggunakan unit LSTM 100 mendapatkan F1-*Score Micro* sebesar 98,5%, F1-*Score Macro* 36,4%, F1-*Score Weighted* sebesar 98,5% dan F1-*Score* rata-rata sebesar 77,8% sedangkan model meggunakan unit LSTM 128 mendapatkan F1-*Score Micro* sebesar 98,5%, F1-*Score Macro* sebesar 36%, F1-*Score Weighted* sebesar 98,5% dan F1-*Score* rata-rata sebesar 77,7%. Dari hasil yang didadaptkan, penggunaan unit LSTM yang lebih sedikit memiliki hasil F1-*Score* yang lebih bagus. Perbedaan antara jumlah unit LSTM 100 dan 128 sebesar 0.1%. Oleh karena itu, pada skenario selanjutnya jumlah unit yang digunakan yaitu 100.

Tabel 4.2 Hasil evaluasi skenario 1 model BiLSTM

Skenario	***Unit LSTM***	**F1-*Score***			**F1-*Score* rata-rata**
		Micro	***Macro***	***Weighted***	
1	100	98,5%	36,4%	98,5%	77,8%
	128	98,5%	36%	98,5%	77,7%

b. Skenario 2

Pada skenario 2, dilakukan perbandingan *learning rate*. Hasil dari evaluasi dapat dilihat pada Tabel 4.3. Dapat diketahui pada hasil evaluasi, dari ketiga *learning rate* yang diuji ternyata *learning rate* yang memiliki nilai F1-*Score* terbaik adalah *learning rate* dengan nilai 0,001. *Learning rate* 0,01 memperoleh F1-*Score Micro* sebesar 98,5%, F1-*Score Macro* sebesar 36,2%, F1-*Score Weighted* sebesar 98,5% dan F1-*Score* rata-rata sebesar 77,8%. *Learning rate* 0,001 memperoleh F1-*Score Micro* sebesar 98,6%, F1-*Score Macro* sebesar 36,4%, F1-*Score Weighted* sebesar 98,6% dan F1-*Score* rata-rata sebesar 77,9%. *Learning rate* 0,0001 memperoleh F1-*Score Micro* sebesar 98,5%, F1-*Score Macro* sebesar 35,4%, F1-*Score Weighted* sebesar 98,5% dan F1-*Score* rata-rata sebesar 77,5%. Oleh karena itu, pada skenario berikutnya *learning rate* yang digunakan yaitu 0,01.

Tabel 4.3 Hasil evaluasi skenario 2 model BiLSTM

Skenario	***Learning rate***	**F1-*Score***			**F1-*Score* rata-rata**
		Micro	***Macro***	***Weighted***	
2	0,01	98,5%	36,2%	98,5%	77,8%
	0,001	98,6%	36,4%	98,6%	77,9%
	0,0001	98,5%	35,4%	98,5%	77,5%

c. Skenario 3

Pada skenario 3, dilakukan perbandingan fungsi optimasi yang digunakan pada model. Hasil evaluasi skenario 3 dapat dilihati di Tabel 4.4. Dari hasil tersebut, dapat diketahui bahwa fungsi optimasi *Adam* memperoleh F1-*Score Micro* sebesar 98,5%, F1-*Score Macro* sebesar 36,3%, F1-*Score Weighted* sebesar 98,5% dan F1-*Score* rata-rata sebesar 77,8%. Fungsi optimasi *Nadam* memperoleh F1-*Score Micro* sebesar 98,6%, F1-*Score Macro* sebesar 36,2%,

F1-*Score Weighted* sebesar 98,6% dan F1-*Score* rata-rata sebesar 77,8%. Hasil evaluasi tersebut menunjukkan bahwa penggunaan fungsi optimasi *Adam* dan *Nadam* memiliki hasil yang sama.

Tabel 4.4 Hasil evaluasi skenario 3 model BiLSTM

Skenario	Fungsi Optimasi	F1-*Score*			F1-*Score* rata-rata
		Micro	*Macro*	*Weighted*	
3	Adam	98,5%	36,3%	98,5%	77,8%
	Nadam	98,6%	36,2%	98,6%	77,8%

Setelah melakukan perbandingan skenario, didapatkan *hyperparameter* terbaik yang menghasilkan model dengan F1-*Score* rata-rata terbaik, yaitu 77,9%. Model tersebut menggunakan *learning rate* sebesar 0,01, unit LSTM sebesar 100, *epoch* sebanyak 30, *batch size* 32, fungsi optimasi *Adam*, algoritma *Word2Vec* berupa *Skip-gram* dan *dropout* sebesar 0,5.

Model BERT

a. Skenario 1

Pada skenario 1 model BERT, dilakukan perbandingan jumlah *epoch*. Hasil evaluasi dapatdilihat pada Tabel 4.5. Dapat diketahui pada hasil evaluasi, dari ketiga *epoch* yang diuji, jumlah *epoch* yang memiliki F1-*Score* terbaik adalah jumlah *epoch* 20. Jumlah *epoch* 4 memperoleh F1-*Score* yang sama pada *Micro, Macro, Weighted* dan rata-rata yaitu 49%. Jumlah *epoch* 10 memperoleh F1-*Score* yang sama pada *Micro, Macro, Weighted* dan rata-rata yaitu 78,5% sedangkan jumlah *epoch* 20 memperoleh F1-*Score* yang sama pada *Micro, Macro, Weighted* dan rata-rata yaitu sebesar 90,3%. Dapat dilihat bahwa penggunaan *epoch* yang lebih besar mendapatkan hasil yang lebih baik, hal ini dikarenakan semakin banyaknya *epoch* maka semakin banyak jumlah perulangan pelatihan dari model. Semakin banyak perulangan pelatihan dari model, maka kemungkinan untuk menemukan performa yang optimal semakin besar. Oleh karena itu, pada skenario berikutnya akan menggunakan jumlah *epoch* 20.

Tabel 4.5 Hasil evaluasi skenario 1 model BERT

Skenario	***Epoch***	**F1-*Score***			**F1-*Score* rata-rata**
		Micro	***Macro***	***Weighted***	
1	4	49%	49%	49%	49%
	10	78,5%	78,5%	78,5%	78,5%
	20	90,3%	90,3%	90,3%	90,3%

b. Skenario 2

Pada skenario 2 model BERT, dilakukan perbandingan *batch size*. Hasil evaluasi dapat dilihat pada Tabel 4.6. Dapat diketahui bahwa, model dengan *batch size* 32 mendapatkan F1-*Score* yang sama pada *Micro, Macro, Weighted* dan rata-rata yaitu sebesar 87,1% sedangkan model dengan *batch size* 64 mendapatkan F1-*Score* yang sama pada *Micro, Macro, Weighted* dan rata-rata yaitu 26,6%. Dapat dilihat dari hasil evaluasi bahwa model dengan *batch size* 32 memiliki hasil yang lebih baik dibanding model dengan menggunakan *batch size* 64. Oleh karena itu, pada skenario berikutnya *batch size* yang digunakan adalah 32.

Tabel 4.6 Hasil evaluasi skenario 2 model BERT

Skenario	**Batch Size**	**F1-*Score***			**F1-*Score* rata-rata**
		Micro	***Macro***	***Weighted***	
2	32	87,1%	87,1%	87,1%	87,1%
	64	26,6%	26,6%	26,6%	26,6%

c. Skenario 3

Pada skenario 3, dilakukan perbandingan nilai *dropout*. Hasil evaluasi dapat dilihat pada Tabel 4.7 . Dapat diketahui bahwa, model tanpa menggunakan *dropout* mendapatkan F1-*Score*yang sama pada *Micro, Macro, Weighted* dan rata-rata yaitu sebesar 94,4%. Model menggunakan nilai *dropout* 0,25 mendapatkan F1-*Score* yang sama pada *Micro, Macro, Weighted* dan rata-rata sebesar 94,4% sedangkan model menggunakan nilai *dropout* 0,5 mendapatkan F1-*Score* yang sama pada *Micro, Macro, Weighted* dan rata-rata sebesar 87,5%. Dapat dilihat dari hasil evaluasi bahwa model tidak menggunakan *dropout* dan model

menggunakan nilai *dropout* 0,25 mendapatkan hasil yang lebih baik dibanding model dengan model menggunakan nilai *dropout* 0,5.

Tabel 4.7 Hasil evaluasi skenario 3 model BERT

Skenario	Dropout	F1-*Score*			F1-*Score* rata-rata
		Micro	*Macro*	*Weighted*	
3	-	94,4%	94,4%	94,4%	94,4%
	0,25	94,4%	94,4%	94,4%	94,4%
	0,50	87,5%	87,5%	87,5%	87,5%

Setelah dilakukan perbandingan skenario, didapatkan konfigurasi *hyperparameter* terbaik yang menghasilkan model dengan rata-rata F1-*Score* terbaik, yaitu 94,4%. Model tersebut menggunakan fungsi optimasi *AdamW, dropout* sebesar 0,25, *batch size* 32 dan jumlah *epoch* 20.

4.8 Deteksi Entitas

Berdasarkan perbandingan pada skenario pada model BiLSTM dan model BERT yang telah dilakukan sebelumnya, model BiLSTM yang menghasilkan kinerja terbaik menggunakan parameter *learning rate* sebesar 0,01, unit LSTM sebesar 100, *epoch* sebanyak 30, *batch size* 32, fungsi optimasi *Adam*, algoritma *Word2Vec* berupa *Skip-gram* dan *dropout* sebesar 0,5 menghasilkan nilai F1-*Score* 77,9%. Model BERT yang menghasilkan kinerja terbaik menggunakan parameter fungsi optimasi *AdamW, dropout* sebesar 0,25, *batch size* 32 dan jumlah *epoch* 20 menghasilkan nilai F1-*Score* 94,4%. Dari model yang memiliki parameter dan hasil kinerja terbaik akan diimplementasikan untuk melakukan deteksi entitas pada resep makanan. Pengujian untuk model dengan parameter dan kinerja terbaik dilakukan sekali. Entitas yang akan dideteksi terdiri dari beberapa kategori, yaitu nama bahan masakan, bumbu masakan dan alat sedangkan untuk kata-kata yang tidak termasuk dalam kategori tersebut akan masuk ke dalamkategori *outside*. Hasil dari pendeteksi entitas menggunakan model BiLSTM yang memiliki F1-*Score* terbaik ditunjukkan pada Gambar 4.18. Hasil dari pendeteksi entitas menggunakan model BERT yang memiliki F1-*Score* terbaik ditunjukkan pada Gambar 4.19.

Word	True	Pred
1	O	O
2	O	O
kg	O	O
udang	B-BAH	B-BAH
segar	O	O
1	O	O
papan	O	O
tahu	B-BAH	B-BAH
cina	I-BAH	I-BAH
atau	O	O
tahu	O	B-BAH
jenis	O	O
lain	O	O
secukupnya	O	O
Gula	B-BUM	B-BUM
dan	O	O
garam	B-BUM	B-BUM
3	O	O
sdm	O	O
minyak	B-BAH	B-BAH
goreng	I-BAH	I-BAH
Bumbu	O	O
halus	O	O
7	O	O
cabe	B-BUM	B-BUM
merah	I-BUM	I-BUM
bawang	B-BUM	B-BUM
putih	I-BUM	I-BUM
secukupnya	O	O
garam	B-BUM	B-BUM
merica	B-BUM	B-BUM
gula	B-BUM	B-BUM
5	O	O
butir	O	O
telur	B-BAH	B-BAH
puyuh	I-BAH	I-BAH
mentah	O	O
2	O	O
batang	O	O
daun	B-BUM	B-BUM
bawang	I-BUM	I-BUM
iris	O	O
tipis	O	O
secukupnya	O	O
air	B-BAH	B-BAH
es	I-BAH	I-BAH
air	B-BAH	B-BAH
dalam	O	O
panci	B-ALT	O
untuk	O	O
merebus	O	O
bakso	B-BAH	B-BAH

Gambar 4.18 Hasil deteksi entitas model BiLSTM

O	[CLS]
O	200
O	gram
B-BAH	udang
O	1
O	kilo
B-BAH	ayam
I-BAH	paha
I-BAH	atas
O	1
O	siung
B-BUM	bawang
I-BUM	merah
O	5
O	siung
B-BUM	bawang
I-BUM	putih
B-BUM	cengkeh
O	1
B-ALT	sendok
I-ALT	makan
B-BUM	gula
O	meruca
B-BUM	garam
O	12
B-ALT	tusuk
I-ALT	sate
O	1
B-ALT	gelas
B-BAH	air
B-BUM	[SEP]

Gambar 4.19 Hasil deteksi entitas model BERT

Dari gambar tersebut, dapat terlihat bahwa dari hasil deteksi yang sudah dilakukan oleh model BiLSTM dan model BERT terbaik, entitas-entitas dapat diprediksi dengan bener walaupun belum sempurna. Gambar 4.17 menunjukkan bahwa entitas dapat diprediksi dengan benar. Pada Gambar 4.16 menunjukkan bahwa entitas dapat diprediksi dengan bener seperti kata “tahu” diprediksi oleh model bahwa kata tersebut adalah bahan masakan. Akan tetapi, dapat ditemukan beberapa kesalahan dalam memprediksi entitas seperti yang ditunjukkan pada Gambar 4.16, bahwa kata “panci” yang seharusnya masuk dalam kategori entitas alat, namun justru tergolong sebagai *outside*. Hal ini disebabkan karena performa model BiLSTM masih perlu ditingkatkan lagi ketika melakukan proses pembelajaran. Sedikitnya jumlah entitas dapat mempengaruhi kinerja model dalam proses pembelajaran, seperti pada kategori label B-ALT dan I-ALT, sehingga akan menjadi sulit dalam memprediksi.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan penelitian *named entity recognition* pada resep masakan menggunakan metode BiLSTM dan BERT, dapat disimpulkan bahwa:

a. Menggunakan NER dapat membantu mengidentifikasi entitas pada resep masakan. NER dapat mengenali entitas penting yang ada dalam resep masakan seperti bahan makanan, bumbu dan alat. Model NER dibangun dengan metode BiLSTM dan BERT. Metode BiLSTM dapat memperoleh informasi masa lalu dan masa depan, sehingga dalam mengenali entitas akan memperhatikan konteks dari sebelumnya dan sesudahnya. Metode BERT merupakan *language model representations* yang bertujuan untuk memprediksi hubungan antar kata dengan menganalisisnya secara keseluruhan, sehingga dalam mengenali entitas akan mempelajari teks tidak berlabel dari sisi kiri dan kanan dengan mengkondisikan konteks di semua lapisan dan mempelajari hubungan kontekstual antara kata-kata dalam teks.
b. Hasil evaluasi model NER menggunakan metode BiLSTM dengan parameter learning rate sebesar 0,01, unit LSTM sebesar 100, epoch sebanyak 30, batch size 32, fungsi optimasi Adam, algoritma Word2Vec berupa Skip-gram dan dropout sebesar 0,5 menghasikan F1-Score 77,9% dalam mengidentifikasi entitas dan model NER meggunakan metode BERT dengan parameter fungsi optimasi AdamW, dropout sebesar 0,25, batch size 32 dan jumlah epoch 20 menghasilkan F1-Score 94,4% dalam mengidentifikasi entitas. Dari hasil F1-Score yang telah didapatkan, dapat dilihat bahwa model NER menggunakan metode BERT memiliki F1-Score yang lebih baik dari model NER menggunakan metode BiLSTM.

5.2 Saran

Berdasarkan penlitian ini, peneliti berharap agar penelitian ini dikembangkan lagi lebih lanjut dengan beberapa saran berikut:

a. Menambahkan jenis entitas pada *dataset* dengan cara melabeli ulang data dengan entitas tambahan selain bahan, bumbu dan alat seperti takaran, bumbu kering, bumbu cair dan sebagainya.
b. Menggunakan data berbahasa selain Bahasa Indonesia seperti Bahasa Inggris agar dapat mendeteksi entitas pada resep Bahasa Inggris.
c. Menambah jumlah *data*, sehingga entitas yang jumlah datanya masih sedikit seperti dalam penelitian ini yaitu entitas Alat bisa dikenali dengan lebih baik.

DAFTAR PUSTAKA

Chantrapornchai, C., & Tunsakul, A. (2021). Information extraction on tourism domain using SpaCy and BERT. *ECTI Transactions on Computer and Information Technology*, *15*(1), 108–122. https://doi.org/10.37936/ecti-cit.2021151.228621

Deng, L., & Liu, Y. (2018). *Deep learning in natural language processing*. Springer.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*(Mlm), 4171–4186.

Diwan, N., Batra, D., & Bagler, G. (2020). A named entity based approach to model recipes. *Proceedings - 2020 IEEE 36th International Conference on Data Engineering Workshops, ICDEW 2020*, 88–93. https://doi.org/10.1109/ICDEW49219.2020.000-2

Eisenstein, J. (2019). *Introduction to natural language processing*. MIT press.

Elnemr, H., Zayed, N., & Fakhreldein, M. (2016). *Feature extraction techniques: fundamental concepts and survey*.

Fimoza, D. (2021). *Analisis Sentimen Terhadap Film Indonesia dengan Pendekatan Bert*.

Gunawan, W., Suhartono, D., Purnomo, F., & Ongko, A. (2018). Named-Entity Recognition for Indonesian Language using Bidirectional LSTM-CNNs. *Procedia Computer Science*, *135*, 425–432. https://doi.org/10.1016/j.procs.2018.08.193

Helwe, C., Dib, G., Shamas, M., & Elbassuoni, S. (2020). A Semi-Supervised BERT Approach for Arabic Named Entity Recognition. *Proceedings of the Fifth Arabic Natural Language Processing Workshop*, 49–57. https://www.aclweb.org/anthology/2020.wanlp-1.5

Huang, Y. A., You, Z. H., Chen, X., Chan, K., & Luo, X. (2016). Sequence-based prediction of proteinprotein interactions using weighted sparse representation model combined with global encoding. *BMC Bioinformatics*, *17*(1), 1–11. https://doi.org/10.1186/s12859-016-1035-4

Kågebäck, M., & Salomonsson, H. (2016). *Word Sense Disambiguation using a Bidirectional LSTM*. http://arxiv.org/abs/1606.03568

Ma, L., & Zhang, Y. (2015). Using Word2Vec to process big text data. *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015, October 2015*, 2895–2897. https://doi.org/10.1109/BigData.2015.7364114

Ma, X., & Hovy, E. (2016). End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, *2*, 1064–1074. https://doi.org/10.18653/v1/p16-1101

Mahalakshmi, G. S., Sreedhar, M. N., Selvam, R. K., & Sendhilkumar, S. (2020). Exploiting

Bi-LSTMs for Named Entity Recognition in Indian Culinary Science. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3545088

Manaswi, N. K., Manaswi, N. K., & John, S. (2018). *Deep learning with applications using python*. Springer.

Mansouri, A., Affendey, L. S., & Mamat, A. (2008). Named Entity Recognition Approaches. *Journal of Computer Science*, *8*(2), 339–344.

Mikolov, T., Le, Q. V., & Sutskever, I. (2013). *Exploiting Similarities among Languages for Machine Translation*. http://arxiv.org/abs/1309.4168

Permana, H., & Purnamasari, K. K. (2019). Named Entity Recognition Using Bidirectional Lstm-Crf Methods in Indonesian Text. *Procedia Computer Science*, *112*.

Reimers, N., & Gurevych, I. (2017). *Optimal Hyperparameters for Deep LSTM-Networks for Sequence Labeling Tasks*. http://arxiv.org/abs/1707.06799

Saputra, A. chandra. (2020). Penentuan Parameter Learning Rate Selama Pembelajaran Jaringan Syaraf Tiruan Backpropagation Menggunakan Algoritma Genetika. *Jurnal Teknologi Informasi: Jurnal Keilmuan Dan Aplikasi Bidang Teknik Informatika*, *14*(2), 202–212. https://doi.org/10.47111/jti.v14i2.1141

Seok, M., Song, H.-J., Park, C.-Y., Kim, J.-D., & Kim, Y. (2015). *Comparison of NER Performance Using Word Embedding*. *May*, 784–788. https://doi.org/10.14257/astl.2015.120.154

Seok, M., Song, H. J., Park, C. Y., Kim, J. D., & Kim, Y. seop. (2016). Named entity recognition using word embedding as a feature. *International Journal of Software Engineering and Its Applications*, *10*(2), 93–104. https://doi.org/10.14257/ijseia.2016.10.2.08

Souza, F., Nogueira, R., & Lotufo, R. (2019). *Portuguese Named Entity Recognition using BERT-CRF*. http://arxiv.org/abs/1909.10649

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

Taher, E., Hoseini, S. A., & Shamsfard, M. (2019). *Beheshti-NER: Persian named entity recognition Using BERT*.

Wang, Q., Zhou, Y., Ruan, T., Gao, D., Xia, Y., & He, P. (2019). Incorporating dictionaries into deep neural networks for the Chinese clinical named entity recognition. *Journal of Biomedical Informatics*, *92*(April 2018), 103133. https://doi.org/10.1016/j.jbi.2019.103133

Wibisono, Y., & Khodra, M. L. (2018). *Pengenalan Entitas Bernama Otomatis untuk Bahasa Indonesia dengan Pendekatan Pembelajaran Mesin*. https://doi.org/10.31227/osf.io/vud2p

LAMPIRAN