# PEMODELAN *NAMED ENTITY RECOGNITION* PADA ARTIKEL WISATA DENGAN METODE *BIDIRECTIONAL LONG SHORT-TERM MEMORY* DAN *CONDITIONAL RANDOM FIELDS*


Disusun Oleh:

N a m a : Annisa Zahra
NIM : 17523075

**PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA**
**FAKULTAS TEKNOLOGI INDUSTRI**
**UNIVERSITAS ISLAM INDONESIA**
**2021**

**HALAMAN PENGESAHAN DOSEN PEMBIMBING**

# PEMODELAN *NAMED ENTITY RECOGNITION* PADA ARTIKEL WISATA DENGAN METODE *BIDIRECTIONAL LONG SHORT-TERM MEMORY* DAN *CONDITIONAL RANDOM FIELDS*

**TUGAS AKHIR**

Disusun Oleh:


N a m a : Annisa Zahra

NIM : 17523075


Yogyakarta, 03 Mei 2021

Pembimbing 1,

( Ahmad Fathan Hidayatullah, S.T., M.Cs. )

Pembimbing 2,

( Septia Rani, S.T., M.Cs. )

**HALAMAN PENGESAHAN DOSEN PENGUJI**

# PEMODELAN *NAMED ENTITY RECOGNITION* PADA ARTIKEL WISATA DENGAN METODE *BIDIRECTIONAL LONG SHORT-TERM MEMORY* DAN *CONDITIONAL RANDOM FIELDS*

## TUGAS AKHIR

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

Yogyakarta, 03 Mei 2021

Tim Penguji

Ahmad Fathan Hidayatullah, S.T., M.Cs. _____

**Anggota 1**

Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. _____

**Anggota 2**

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

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 : Annisa Zahra

NIM : 17523075

Tugas akhir dengan judul:

# PEMODELAN *NAMED ENTITY RECOGNITION* PADA ARTIKEL WISATA DENGAN METODE *BIDIRECTIONAL LONG SHORT-TERM MEMORY* DAN *CONDITIONAL RANDOM FIELDS*

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, 03 Mei 2021


( Annisa Zahra )

# HALAMAN PERSEMBAHAN

Tugas Akhir ini saya persembahkan untuk kedua orang tua dan kedua adik saya tersayang, keluarga, teman, serta seluruh pihak yang sudah mendukung, membantu, dan mendo'akan selama proses pengerjaannya hingga akhirnya selesai, juga untuk diri saya sendiri yang sudah menikmati segala prosesnya.

## HALAMAN MOTO

"*Be kind, for whenever kindness becomes part of something, it beautifies it. Whenever it is taken from something, it leaves it tarnished.*"

**— Nabi Muhammad SAW**

"*Failure is the condiment that gives success its flavor.*"

**— Truman Capote**

# KATA PENGANTAR

*Assalamu'alaikum Warahmatullahi Wabarakatuh.*

*Alhamdulillahirabbil'alamin,* segala puji bagi Allah SWT yang telah memberikan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir mengenai "PEMODELAN *NAMED ENTITY RECOGNITION* PADA ARTIKEL WISATA DENGAN METODE *BIDIRECTIONAL LONG SHORT-TERM MEMORY* DAN *CONDITIONAL RANDOM FIELDS*" guna memenuhi syarat untuk menyelesaikan pendidikan pada jenjang sarjana. Penyelesaian tugas akhir ini tidak lepas dari beberapa kendala, namun, berkat segala doa, bimbingan, motivasi, serta semangat yang tiada henti diberikan orang-orang di sekitar penulis, kendala tersebut dapat dilalui. Sehingga pada kesempatan ini penulis ingin berterima kasih kepada:

1. Kedua orang tua penulis yang selalu memberikan do'a, dukungan, motivasi, serta kasih sayang yang tiada henti kepada penulis.
2. Kedua adik penulis yang sangat pengertian dan selalu memberikan dukungan.
3. Seluruh keluarga yang juga senantiasa memberikan *support*.
4. Bapak Prof. Fathul Wahid, S.T., M.Sc., Ph.D. selaku Rektor Universitas Islam Indonesia.
5. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc. selaku Kaprodi Informatika UII.
6. Bapak Ahmad Fathan Hidayatullah, S.T., M.Cs. dan Ibu Septia Rani, S.T., M.Cs. selaku dosen pembimbing yang sudah bersedia meluangkan waktu dan tenaga dalam memberikan ilmu, arahan, serta bimbingan kepada penulis untuk menyelesaikan tugas akhir ini dengan baik.
7. Seluruh dosen Informatika UII yang telah memberikan ilmu, juga nasihat yang sangat berarti dan bermanfaat, semoga Allah SWT senantiasa melindungi dan membalas jasa, serta kebaikan Bapak/Ibu sekalian.
8. Aldhiyatika Amwin, Annisa Nauli Hasibuan, dan M. Zikri Khatami Sagala alias Gozy selaku teman-teman yang senantiasa menghiasi hari-hari penulis selama masa kuliah.
9. Nabila Annisa Haque, M. Wahyu Alwi Siregar, dan Mhd. Idris Syahputra selaku teman-teman yang juga sangat berarti bagi penulis sejak SMA.
10. Syarifah Elza Ramadhania, Adelia Sukma Ardana, serta teman-teman PIXEL lainnya yang membuat masa perkuliahan penulis menjadi lebih berwarna.

Penulis menyadari bahwa tugas akhir ini masih memiliki kekurangan dan juga terbuka terhadap kritik serta saran yang ingin disampaikan agar penelitian selanjutnya menjadi lebih baik. Semoga laporan ini dapat berguna dan bermanfaat ke depannya.

*Wassalamu'alaikum Warahmatullahi Wabarakatuh.*

Yogyakarta, 03 Mei 2021

( Annisa Zahra )

# SARI

Pada proses perencanaan perjalanan wisata, calon wisatawan umumnya melakukan pencarian destinasi wisata terlebih dahulu. Pencarian tersebut kerap kali dilakukan melalui internet dengan bantuan mesin pencari, salah satu caranya adalah dengan membaca artikel yang tersedia. Proses menemukan informasi yang relevan pada artikel-artikel tersebut adakalanya membutuhkan waktu yang tidak sedikit karena harus membaca satu per satu artikel.

*Named Entity Recognition* (NER) dapat digunakan untuk mendeteksi entitas tertentu pada suatu teks yang akan membantu pengguna untuk menemukan informasi yang diinginkan. Tujuan penelitian ini adalah membuat pemodelan NER yang akan membantu dalam pendeteksian tempat wisata di suatu artikel. Artikel yang digunakan adalah artikel berbahasa Inggris dari internet. Terdapat 92 artikel yang sudah dikumpulkan dan dipecah menjadi 183.507 token kata. Token-token tersebut selanjutnya diberi label sesuai jenisnya masing-masing. Tempat wisata akan digolongkan menjadi *heritage, natural,* dan *purpose*. Token selain tempat wisata akan diberi label O (*outside*).

Penelitian ini menggunakan *StratifiedKFold* untuk membagi *dataset*, serta ekstraksi fitur menggunakan *word embedding* dengan teknik *Word2Vec,* dan gabungan metode *Bidirectional Long Short-Term Memory* (BiLSTM) dan *Conditional Random Fields* (CRF) pada pemodelannya. Berdasarkan hasil yang diperoleh, model yang dihasilkan mampu mendeteksi beberapa entitas tempat wisata, namun dalam pendeteksiannya masih ditemukan banyak kesalahan. Dari beberapa skenario model yang diuji, rata-rata F1-*Score* tertinggi yang dihasilkan sebesar 75,25%.

Kata kunci: BiLSTM, CRF, ekstraksi fitur, *Named Entity Recognition, StratifiedKFold,* wisata, *Word2Vec*.

# GLOSARIUM

| | |
|---|---|
| Bobot | parameter pada *neural network* yang mentransformasikan data masukan dalam lapisan tersembunyi jaringan. |
| *Dataset* | kumpulan data yang digunakan saat membangun model dan fitur. |
| *Hyperparameter* | variabel konfigurasi yang bersifat eksternal pada model dan nilainya tidak dapat diperkirakan dari data. |
| *List* | tipe data pada bahasa pemrograman *Python* yang berada di antara dua kurung siku dan digunakan untuk menyimpan suatu daftar nilai yang dipisahkan dengan koma. |
| *Neural network* | rangkaian algoritma yang berusaha mengenali hubungan yang mendasari sekumpulan data melalui proses yang meniru cara kerja otak manusia. |
| *Node* | sebuah unit komputasi yang memiliki satu atau lebih koneksi masukan berbobot, sebuah fungsi transfer yang menggabungkan masukan dalam beberapa cara, dan juga sebuah koneksi keluaran. |
| Parameter | variabel konfigurasi yang bersifat internal pada model dan nilainya dapat diperkirakan dari data. |
| *Tuple* | tipe data pada bahasa pemrograman *Python* yang digunakan untuk menyimpan berbagai tipe data dan isinya tidak dapat diubah. |

# DAFTAR ISI

# DAFTAR TABEL

# DAFTAR GAMBAR

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

Pertumbuhan pariwisata terjadi setiap tahunnya. Pada tahun 2019, tercatat sebanyak 1,5 miliar kunjungan wisatawan internasional secara global. Hal tersebut mengalami peningkatan sebesar 4% dari tahun sebelumnya (UNWTO, 2020). Salah satu studi yang dilakukan oleh *Google Travel* menemukan bahwa 74% wisatawan merencanakan perjalanan mereka melalui internet ("The 2014 Traveler's Road to Decision," 2014). Pencarian destinasi wisata adalah salah satu tahapan yang umumnya dilakukan pada saat merencanakan perjalanan. Proses pencarian tersebut dapat dilakukan dengan membaca artikel yang beredar. Mengumpulkan informasi pariwisata dengan cara membaca buku panduan perjalanan dan situs portal yang disediakan oleh perusahaan wisata dan dewan wisata pemerintah akan memakan waktu, begitu juga jika harus menelusuri hasil dari mesin pencari, memilih, dan melihat detail masing-masing akomodasi (Chantrapornchai & Tunsakul, 2019; Ishino, Nanba, & Takezawa, 2011).

Dengan demikian, untuk memudahkan calon wisatawan dalam memperoleh informasi wisata, perlu dilakukan penelitian untuk mendeteksi entitas wisata pada suatu teks yang akan berguna pada saat ekstraksi informasi wisata ke depannya. Hasil dari proses ekstraksi tidak hanya dapat berguna bagi calon wisatawan. Namun, dengan adanya kemajuan dalam ekonomi pariwisata yang telah memungkinkan untuk mengumpulkan sejumlah besar data perjalanan wisata yang jika dianalisis dengan benar, hasilnya dapat membantu dalam pengambilan keputusan secara *real-time* dan juga untuk penyediaan rekomendasi wisata (Ge, Xiong, Tuzhilin, & Liu, 2014). Sebelum melakukan analisis, dibutuhkan pengambilan informasi yang diperlukan. Kepentingan untuk mengambil informasi yang diperlukan dari internet dengan domain spesifik dapat menyelesaikan masalah dan dapat digunakan untuk pemrosesan *query* yang cepat, pemahaman yang efisien tentang konteks data, memahami pelanggan, meningkatkan bisnis, dan untuk rekomendasi yang dipersonalisasi (Vijay & Sridhar, 2016). Pengambilan informasi dari data yang ingin digunakan dapat dilakukan dengan cara ekstraksi.

*Named Entity Recognition* (NER) dapat membantu proses ekstraksi informasi dengan cara mengidentifikasi suatu entitas nama. Proses NER membantu pengguna untuk menghasilkan korpus yang lebih bermakna dengan mengidentifikasi nama-nama yang tepat di korpus dan mengklasifikasikannya ke dalam kelompok-kelompok seperti orang, organisasi,

lokasi, dan lainnya (Alfred, Leong, On, & Anthony, 2014). Korpus merupakan kumpulan besar data linguistik, baik teks tertulis atau transkripsi ucapan terekam, yang dapat digunakan sebagai titik awal deskripsi linguistik atau sebagai alat verifikasi hipotesis tentang suatu bahasa (Crystal, 1995). Pada domain wisata, entitas yang diidentifikasi dapat berupa nama tempat wisata, tempat penginapan, fasilitas, serta lokasinya. Identifikasi entitas terkait diharapkan dapat memudahkan calon wisatawan dalam menemukan destinasi wisata melalui internet.

Berbagai pendekatan NER adalah *Rule Based, Machine Learning* yang meliputi metode *Hidden Markov Model* (HMM), *Maximum Entropy, Decision Tree, Support Vector Machines* (SVM)*, Conditional Random Fields* (CRF), dan Pendekatan Hibrida (Kaur & Gupta, 2010). Terdapat juga metode lain seperti *Recurrent Neural Network* (RNN) dan variannya, yaitu *Long Short-Term Memory* (LSTM) yang telah berhasil digunakan dalam berbagai masalah prediksi urutan, seperti NER, pemodelan bahasa, dan pengenalan suara (Lyu, Chen, Ren, & Ji, 2017).

Penelitian ini menawarkan solusi untuk mempermudah wisatawan dalam mencari destinasi wisata dari artikel berbahasa Inggris di internet dengan menggunakan teknik NER untuk membantu proses ekstraksi informasi berupa tempat wisata pada suatu artikel. Metode yang digunakan adalah *Bidirectional* LSTM (BiLSTM) dan CRF. Penelitian menggunakan gabungan metode tersebut sebelumnya sudah pernah dilakukan oleh (Z. Huang, Research, Xu, & Baidu, 2015) yang menggunakan 3 *dataset*, yaitu *Penn TreeBank* (PTB) POS *Tagging*, CoNLL 2000 *chunking*, dan CoNLL 2003 *named entity tagging*. Penelitian tersebut menunjukkan bahwa model BiLSTM-CRF dapat secara efisien menggunakan fitur masukan dari masa lalu dan masa depan berkat adanya komponen LSTM dua arah, serta dapat menggunakan informasi mengenai label pada level kalimat berkat lapisan CRF, dan dari beberapa skenario yang digunakan, model BiLSTM-CRF memberikan hasil yang terbaik hampir di semua *dataset*.

## 1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas, rumusan masalah yang diangkat pada penelitian ini adalah:

a. Bagaimana membangun model *Named Entity Recognition* (NER) pada domain wisata menggunakan gabungan metode BiLSTM dan CRF?
b. Bagaimana performa gabungan metode BiLSTM dan CRF dalam mendeteksi entitas tempat wisata pada suatu artikel?

## 1.3 Tujuan Penelitian

Tujuan dari penelitian ini, yaitu:

a. Membangun model *Named Entity Recognition* (NER) pada domain wisata menggunakan gabungan metode BiLSTM dan CRF.
b. Menguji performa gabungan metode BiLSTM dan CRF dalam mendeteksi entitas tempat wisata pada suatu artikel untuk mengetahui F1-*Score* yang dihasilkan.

## 1.4 Batasan Masalah

a. Data yang digunakan hanya artikel mengenai wisata yang berbahasa Inggris.
b. Informasi yang diidentifikasi hanya tempat wisata.
c. Setiap masukan pada model memiliki panjang 90 kata, sehingga kalimat yang terdiri lebih dari 90 kata akan dipotong menjadi 90 kata.

## 1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah:

a. Mengetahui performa gabungan metode BiLSTM dan CRF dalam melakukan *Named Entity Recognition* pada artikel wisata.
b. Membantu dalam mengidentifikasi tempat wisata pada artikel wisata.

## 1.6 Sistematika Penulisan

Adapun sistematika penulisan pada penelitian ini sebagai berikut:

**BAB I PENDAHULUAN**

Bagian pendahuluan membahas mengenai latar belakang dilakukannya penelitian ini, serta rumusan masalah, tujuan penelitian, batasan masalah, dan manfaat penelitian.

**BAB II LANDASAN TEORI**

Bagian ini membahas tentang penelitian sebelumnya dan juga dasar teori yang berkaitan dengan *Named Entity Recognition*.

**BAB III METODOLOGI PENELITIAN**

Bagian ini menjelaskan langkah-langkah yang dilakukan dalam penelitian ini.

**BAB IV HASIL DAN PEMBAHASAN**

Bagian ini menjabarkan hasil dari penelitian mengenai *Named Entity Recognition* pada artikel wisata.

**BAB V KESIMPULAN DAN SARAN**

Bab kesimpulan dan saran menjelaskan kesimpulan yang diperoleh dari penelitian ini dan juga memberikan saran agar penelitian selanjutnya dapat dilakukan dengan lebih baik.

# BAB II
# LANDASAN TEORI

## 2.1 Penelitian Terkait

Bagian ini membahas tentang penelitian terdahulu yang dijadikan referensi pada penelitian ini. Penelitian-penelitian tersebut ditunjukkan pada Tabel 2.1.

Tabel 2.1 Penelitian terkait

| No. | Judul Penelitian | Metode | Hasil |
|---|---|---|---|
| 1 | *Incorporating Dictionaries into Deep Neural Networks for the Chinese Clinical Named Entity Recognition* (Wang et al., 2019) | BiLSTM-CRF | Penggabungan sebuah kamus dengan sebuah model BiLSTM-CRF memperoleh hasil yang lebih baik dibandingkan jika menggunakan model BiLSTM-CRF saja pada saat diuji coba dengan *dataset* berbahasa Cina. |
| 2 | *Conditional Random Field Based Named Entity Recognition In Geological Text* (Sobhana, Mitra, & Ghosh, 2010) | CRF | Mereka mengembangkan sistem NER untuk teks geologi menggunakan metode CRF. *Dataset* yang digunakan adalah IITKGP-GEOCORP yang dikembangkan dari koleksi artikel dan laporan ilmiah. Melalui kombinasi beberapa fitur (*Prefix* dan *Suffix*, informasi tentang kata-kata di sekitarnya, *Part of Speech*, fitur digit, dan *Named Entity Tag*) F1-*Score* yang dihasilkan sebesar 75,8%. |
| 3 | *Named Entity Recognition Using Word Embedding As A Feature* (Seok, Song, Park, Kim, & Kim, 2016) | CRF | Penelitian ini menambahkan *word embedding* sebagai fiturnya dan CRF sebagai algoritma pembelajarannya. Metode *embedding* yang digunakan ada tiga, yaitu *GloVe*, *Word2Vec*, dan CCA. Nilai F1-*Score* tertinggi pada pengujian A diperoleh dengan menggunaan metode CCA, yaitu sebesar 85,96% dan pada pengujian B diperoleh dengan metode *Word2Vec*, yaitu sebesar 80,72%. Penelitian ini juga membutikan bahwa hasil dari penambahan fitur |

| | | | |
|---|---|---|---|
| | | | *word embedding* lebih baik dibandingkan dengan hasil yang tidak menggunakan fitur tersebut. |
| 4 | *Integrating Dictionary Feature into A Deep Learning Model for Disease Named Entity Recognition* (Nayel & Shashirekha, 2019) | BiLSTM-CRF | Setiap kata pada masukan yang menggunakan *dataset* berbahasa Inggris akan direpresentasikan sebagai vektor. Hasil evaluasi terbaik didapat saat vektor tersebut berisi gabungan informasi dari kamus, *character embedding*, dan *word embedding*. |
| 5 | *Named Entity Recognition From Biomedical Texts Using A Fusion Attention-Based* BiLSTM-CRF (Wei et al., 2019) | BiLSTM dan CRF | Pada penelitian ini, penggabungan lapisan BiLSTM dengan CRF secara efektif dapat memecahkan masalah ketidakmampuan untuk menangani ketergantungan yang kuat dari *tag* dalam suatu urutan. Dengan arsitektur yang sederhana pada korpus JNLPBA, model mereka memperoleh F1-*Score* sebesar 73,50%. Model tersebut juga dapat meningkatkan kemampuan jaringan saraf untuk mengekstrak informasi yang signifikan dan tidak bergantung pada rekayasa fitur, dengan hanya melakukan pra-pelatihan umum pada vektor kata, hal itu membuat model mereka memiliki portabilitas dan ekstensibilitas yang tinggi. |
| 6 | *A Method of Chinese Tourism Named Entity Recognition Based on* BBLC *Model* (Xue, Cao, Ye, & Qin, 2019) | BERT-BiLSTM-CRF | Penelitian ini mengombinasikan BERT dengan metode BiLSTM dan CRF yang disebut dengan model BBLC (BERT-BiLSTM-CRF). Model tersebut kemudian diuji coba dengan menggunakan *dataset* yang berisikan data berupa teks pada domain wisata. *Dataset* yang sama juga diuji coba pada model BiLSTM-CRF dan model CRF. BBLC memperoleh nilai F1-*Score* yang lebih tinggi daripada model lainnya pada entitas *location, organization*, dan *thing*. Sementara F1-*Score* tertinggi pada entitas *person* diraih oleh model CRF. Model BiLSTM-CRF meraih nilai F1-*Score* tertinggi pada entitas *time.* |

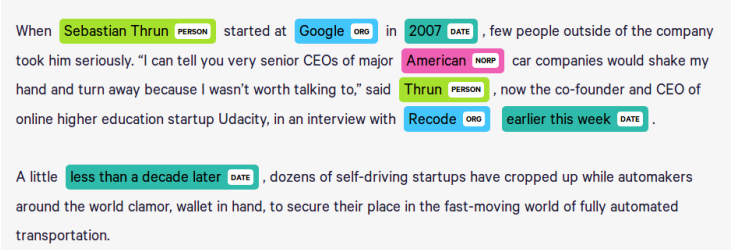
| | | | |
|---|---|---|---|
| 7 | LSTM-CRF *For Drug-Named Entity Recognition* (Zeng, Sun, Lin, & Liu, 2017) | BiLSTM dan CRF | Vektor masukan pada arsitektur penelitian ini merupakan kombinasi dari *word embedding* dengan *character level embedding* dari sebuah kalimat, sedangkan keluarannya berupa urutan label. Hasil penelitian mereka memperoleh nilai F1-*Score* sebesar 92,04%. Usulan gabungan metode BiLSTM dan CRF yang mereka lakukan mengungguli sistem terbaik yang ada pada tantangan DDI2013. |
| 8 | *Bidirectional* LSTM-CRF *Models For Sequence Tagging* (Z. Huang et al., 2015) | BiLSTM dan CRF | Penelitian ini menggunakan beberapa variasi model LSTM untuk *sequence tagging.* Variasi model yang digunakan adalah LSTM, BiLSTM, LSTM-CRF, dan BiLSTM-CRF. Gabungan model BiLSTM dan CRF memperoleh hasil tertinggi pada akurasi *tagging* dan juga merupakan model yang lebih kuat dan tidak terlalu terpengaruh oleh penghapusan fitur-fitur teknik. |

Penelitian yang dilakukan peneliti adalah *Named Entity Recognition* pada domain wisata dengan menggunakan gabungan metode BiLSTM dan CRF. BiLSTM dan CRF dipilih karena dalam mengidentifikasi entitas nama, dibutuhkan pemahaman yang bagus terhadap konteks suatu kata dalam kalimat, BiLSTM dengan keunggulannya yang bisa menyimpan informasi dari masa lalu dan masa depan digabungkan dengan CRF yang pada prosesnya sangat memperhatikan label-label di sekitarnya, maka gabungan kedua metode tersebut diharapkan dapat menghasilkan performa yang baik. Data yang digunakan adalah artikel wisata berbahasa Inggris yang diperoleh dari internet dengan teknik *web scraping*. Ekstraksi fitur yang digunakan adalah *word embedding* dengan teknik *Word2Vec*.

## 2.2 Dasar Teori

### 2.2.1 *Named Entity Recognition* (NER)

NER adalah sebuah tugas komputasi linguistik untuk mengklasifikasikan setiap kata dalam dokumen sebagai salah satu dari delapan kategori, yaitu orang, lokasi, organisasi, tanggal, waktu, persentase, nilai moneter, dan bukan dari salah satu yang sudah disebutkan (Borthwick, 1999). NER mendeteksi sebuah entitas terlebih dahulu dan selanjutnya akan menetapkan kategori dari entitas nama tersebut. Gambar 2.1 merupakan contoh dari NER yang diklasifikasikan menjadi empat kategori, yaitu orang (*PERSON)*, organisasi (ORG), tanggal (*DATE*), dan kebangsaan atau agama atau kelompok politik (NORP).




Gambar 2.1 Contoh NER

Sumber: (Chavan, 2019)

Pengaplikasian NER dapat diterapkan dalam berbagai kasus, seperti membangun mesin pencari di internet yang lebih akurat, pengindeksan buku secara otomatis, pemberian *named-entity tag* yang selanjutnya dapat berfungsi sebagai langkah *preprocessing* untuk menyederhanakan tugas-tugas seperti *machine translation*. *Named-entity tagger* juga merupakan komponen penting dari tugas ekstraksi informasi yang lebih kompleks (Borthwick, 1999).

### 2.2.2 *Web Scraping*

*Web scraping* adalah proses mengekstrak data dari web secara terprogram dan mengubahnya menjadi kumpulan data terstruktur. Proses tersebut memungkinkan jumlah data yang lebih besar untuk dikumpulkan dalam rentang waktu yang lebih singkat dan dengan cara otomatis yang dapat meminimalkan kesalahan (Dogucu & Çetinkaya-Rundel, 2020). *Web scraping* dapat digunakan pada berbagai macam skenario, seperti *scraping* kontak, perbandingan perubahan harga, pengumpulan ulasan produk, pengumpulan daftar *real estate*, pemantauan data cuaca, deteksi perubahan situs web, dan integrasi data web (Zhao, 2017).

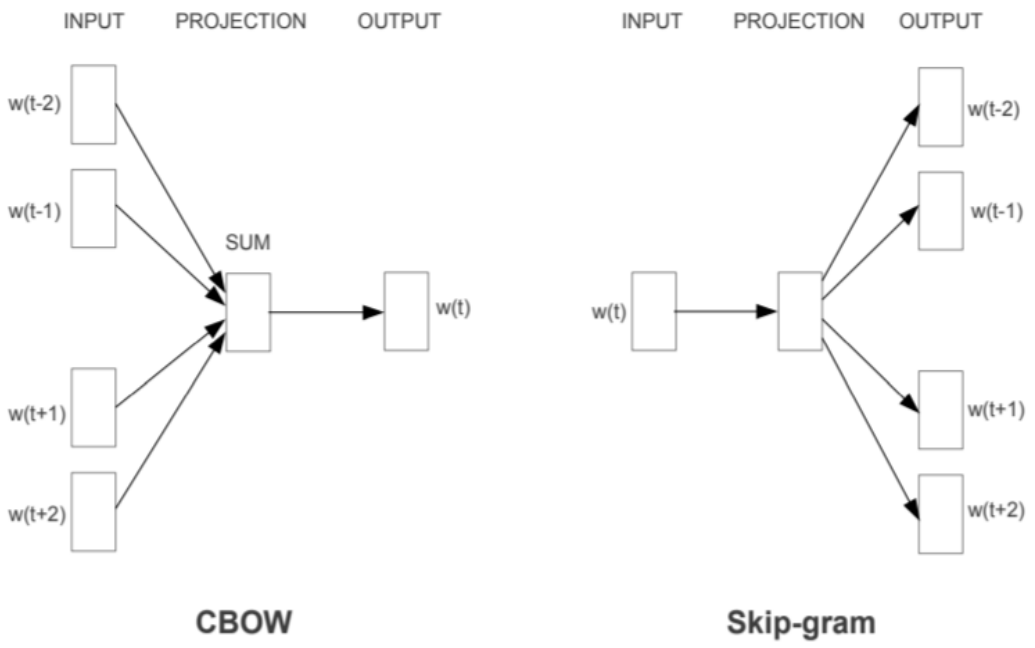
Proses *scraping* data dari internet dapat dimulai dengan memperoleh sumber daya dari web dan kemudian diekstrak untuk mendapatkan informasi yang diinginkan. Ada dua modul penting dari sebuah program *web scraping*, yaitu modul untuk membuat HTTP *request*, seperti Urllib2, dan juga modul untuk melakukan *parsing* serta ekstraksi informasi dari kode HTML mentah, seperti *Newspaper*.

### 2.2.3 Ekstraksi Fitur

Ekstraksi fitur bertujuan untuk menggali informasi yang berguna dari sampel asli dan merepresentasikannya sebagai vektor fitur yang dinormalisasi dengan ukuran yang sama. Metode ekstraksi fitur yang efektif biasanya membantu meningkatkan kinerja sistem prediksi (Huang, You, Chen, Chan, & Luo, 2016). Fitur juga dapat membantu dalam menentukan sebuah entitas masuk ke kategori kelas yang mana (Sobhana et al., 2010). Jenis fitur sendiri ada berbagai macam. Namun, yang digunakan pada penelitian ini adalah *word embedding* dengan teknik *Word2Vec*.

### 2.2.4 *Word2Vec*

*Word2Vec* akan mengubah kata-kata ke dalam bentuk vektor. Terdapat dua algoritma pembelajaran yang utama pada *Word2Vec*, yaitu *Continuous Bag-of-Words* (CBOW) dan *Skip-gram*. Arsitektur kedua algoritma tersebut ditunjukkan pada Gambar 2.2.




Gambar 2.2 Arsitektur CBOW dan *Skip-gram*
Sumber: (Mikolov, Chen, Corrado, & Dean, 2013)

Pada arsitektur CBOW, model akan memprediksi kata saat ini berdasarkan konteks kata di sekitarnya, sedangkan pada arsitektur *Skip-gram*, model menggunaan kata saat ini untuk memprediksi konteks kata di sekitarnya. Tujuan pelatihan model CBOW adalah untuk

menggabungkan representasi kata di sekitarnya untuk memprediksi kata yang ada di tengah, sedangkan tujuan pelatihan model *Skip-gram* adalah untuk mempelajari representasi vektor kata yang baik pada saat memprediksi konteksnya dalam kalimat yang sama, dan karena memiliki kompleksitas komputasi yang rendah, kedua model tersebut dapat dilatih pada korpus yang berukuran besar dalam waktu singkat (Mikolov & Le, n.d.).

### 2.2.5 *Conditional Random Fields* (CRF)

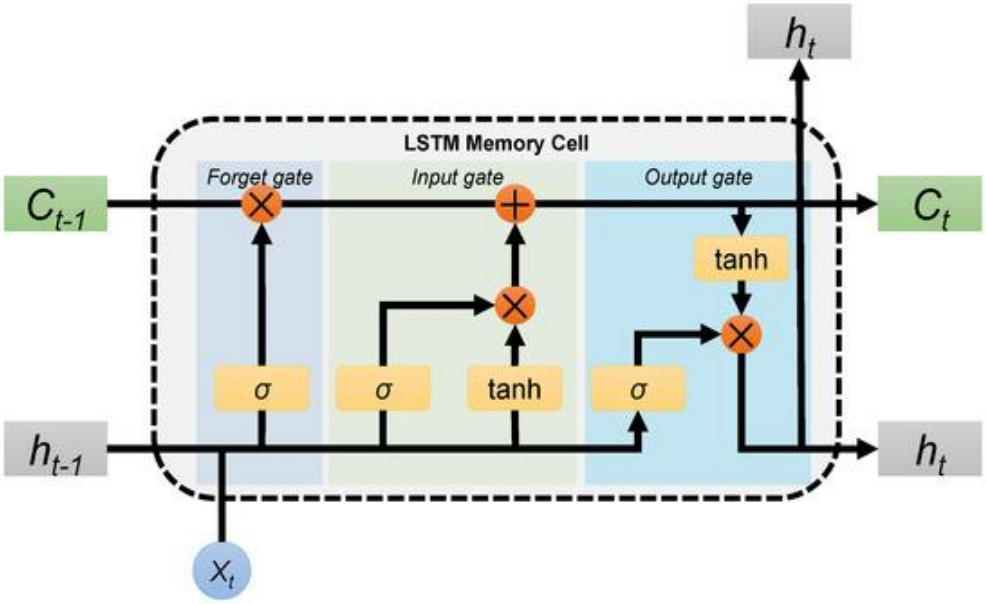
CRF adalah sebuah kerangka kerja yang digunakan dalam membangun model probabilistik untuk melakukan segmentasi dan pemberian label pada data yang berurutan. Selain CRF, terdapat juga kerangka kerja lain, seperti *Hidden Markov Model* (HMM) dan *Maximum Entropy Markov Model* (MEMM) yang dapat digunakan untuk menangani kasus segmentasi dan pelabelan data. Kelebihan CRF dibandingkan dengan HMM, yaitu CRF dapat menangani adanya problem ketergantungan asumsi yang tinggi pada HMM. Sedangkan kelebihan CRF dibandingkan dengan MEMM, yaitu CRF memiliki semua keunggulan dari MEMM dan terlebih lagi CRF juga mampu menangani problem bias label yang terjadi pada MEMM (Lafferty, Mccallum, & Pereira, 2001). Penanganan bias label tersebut dapat terjadi karena CRF tidak bergantung pada asumsi independen yang menyatakan bahwa label tidak bergantung antara satu dengan yang lainnya.

CRF tergolong sebagai model diskriminatif. Model diskriminatif akan memodelkan batasan keputusan antar setiap kelas. Data masukan pada CRF berbentuk sekuensial dan pada dasarnya CRF sendiri merupakan versi sekuensial dari regresi logistik yang menggunakan distribusi probabilitas bersyarat. Namun, untuk CRF algoritmanya diperluas dengan menerapkan fungsi fitur. Tujuan dari penggunaan fungsi fitur itu sendiri adalah untuk mengekspresikan beberapa jenis karakteristik urutan yang diwakili oleh suatu data.

Proses implementasi CRF dapat dilakukan dengan mengekstrak fitur yang dibutuhkan terlebih dahulu. Pemilihan fitur-fitur tersebut akan berdampak pada akurasi dari model yang dibuat sehingga untuk meningkatkan akurasi, maka fitur serta kombinasinya juga harus tepat. Langkah selanjutnya adalah melakukan penaksiran parameter guna mendapatkan nilai parameter fungsi fitur yang optimal. Nilai optimal tersebut dapat diperoleh dengan menggunakan prosedur maksimum *likelihood* yang nantinya akan menunjukkan seberapa banyak parameter yang ada pada data pelatihan. Langkah terakhir adalah penerapan model terhadap data uji (Wilyawan, 2018).

### 2.2.6 *Long Short-Term Memory* (LSTM)

LSTM merupakan modifikasi dari *Recurrent Neural Network* (RNN) dengan adanya penambahan *memory cell* yang digunakan untuk menyimpan informasi dengan jangka waktu yang panjang. LSTM juga dapat menangani masalah *vanishing gradient* yang terdapat pada RNN saat memproses data sekuensial yang panjang dengan menggunakan satu set gerbang yang digunakan untuk mengontrol informasi yang masuk ke memori (Manaswi, 2018)




Gambar 2.3 LSTM *Memory Cell*

Sumber: (Fan et al., 2020)

Gambar 2.3 menunjukkan arsitektur *memory cell* dari LSTM yang terdiri dari tiga gerbang/*gate*. *Forget gate* berguna untuk menentukan informasi mana saja yang akan dibuang atau disimpan dari *hidden state* sebelumnya ($h_{t-1}$) dan informasi dari masukan saat ini ($X_t$). Keputusan tersebut dibuat oleh fungsi aktivasi *sigmoid* yang akan menghasilkan keluaran antara 1 dan 0. Semakin dekat ke angka 1 berarti informasi tersebut akan disimpan, sedangkan jika semakin dekat ke angka 0 berarti informasi tersebut akan dibuang.

*Input gate* memiliki sebuah lapisan *sigmoid* yang berguna untuk menentukan nilai mana yang akan diperbarui dengan mengubah nilai tersebut menjadi antara 0 dan 1. Angka 0 berarti tidak penting, sedangkan angka 1 berarti penting. Sementara lapisan *tanh* akan membuat sebuah vektor yang berisi kandidat nilai baru yang akan ditambahkan ke dalam *memory cell.*

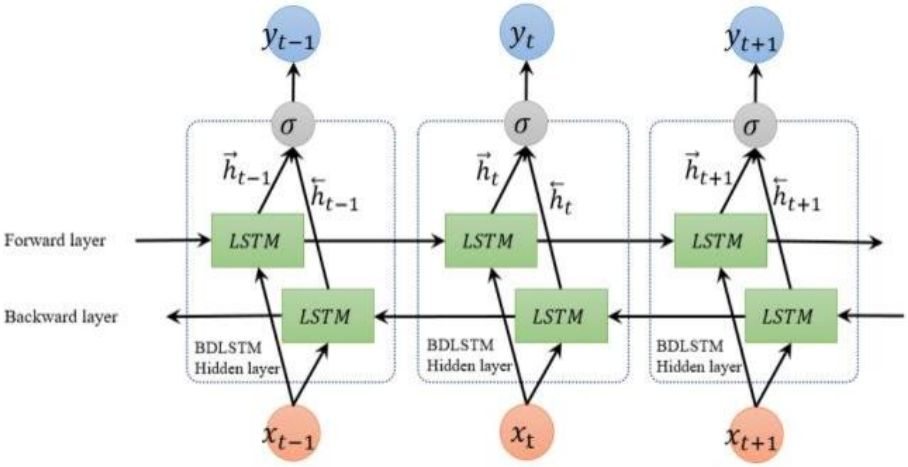
*Output gate* akan menentukan seperti apa keadaan *hidden state* yang selanjutnya. *Hidden state* digunakan untuk prediksi dan mengandung informasi dari masukan sebelumnya.

### 2.2.7 *Bidirectional Long Short-Term Memory* (BiLSTM)

LSTM akan sangat bermanfaat dalam hal pelabelan sekuensial apabila memiliki akses terhadap kedua informasi dari sebelum dan sesudahnya. Namun, *hidden state* pada LSTM hanya mengambil informasi dari sebelumnya (masa lalu), sedangkan untuk informasi yang ada setelahnya tidak diketahui. Permasalahan tersebut dapat dipecahkan dengan menggunakan BiLSTM (Ma & Hovy, n.d.). Pada dasarnya BiLSTM terdiri dari dua LSTM, *forward* LSTM dan *backward* LSTM, sehingga gabungan tersebut akan menangkap informasi dari kedua arah. Gambar 2.4 menunjukkan arsitektur dari BiLSTM. Arsitektur tersebut menunjukkan adanya dua *layer* LSTM pada *output layer* yang sama sehingga juga terdapat dua *hidden layer*, yaitu $\overrightarrow{\boldsymbol{h}}$ dan $\overleftarrow{\boldsymbol{h}}$. Urutan keluaran pada *forward layer*, $\overrightarrow{\boldsymbol{h}}$ dihitung secara berulang menggunakan masukan dalam urutan yang positif dari waktu $t-1$ ke waktu $t+1$, sedangkan pada *backward layer,* $\overleftarrow{\boldsymbol{h}}$ dihitung menggunakan masukan yang terbalik dari waktu $t+1$ ke waktu $t-1$. BiLSTM *layer* menghasilkan vektor keluaran, $\boldsymbol{Y_t}$, yang mana setiap elemen akan dihitung dengan menggunakan Persamaan ( 2.1 ).

$$y_t = \sigma(\overrightarrow{h}, \overleftarrow{h}) \qquad (2.1)$$

Fungsi σ digunakan untuk mengombinasikan dua urutan keluaran. Hal tersebut dapat berupa fungsi penggabungan, fungsi penjumlahan, fungsi rata-rata atau fungsi perkalian.



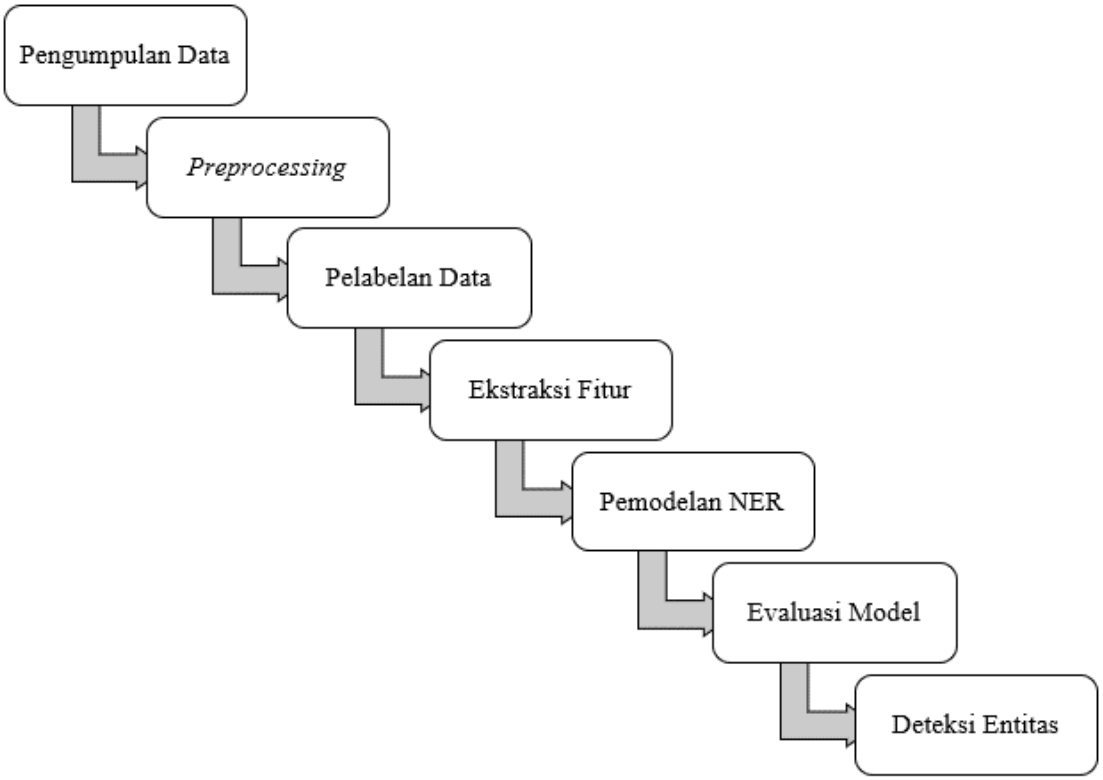

Gambar 2.4 Arsitektur BiLSTM

Sumber: (Cui, Ke, & Wang, 2018)

# BAB III
# METODOLOGI PENELITIAN

## 3.1 Langkah Pengerjaan Tugas Akhir

Gambar 3.1 menunjukkan langkah-langkah yang digunakan dalam mengerjakan penelitian ini, yaitu pengumpulan data, *preprocessing*, pelabelan data, ekstraksi fitur, pemodelan NER, evaluasi model, dan deteksi entitas.



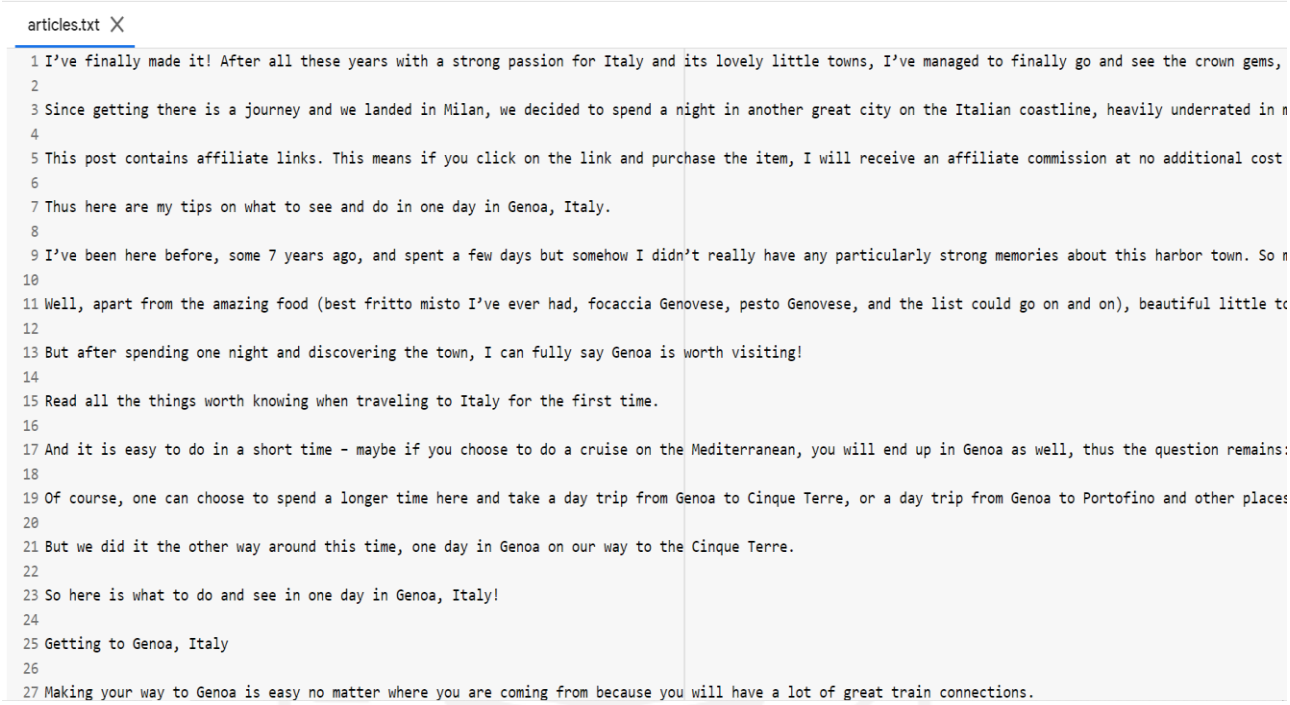

Gambar 3.1 Langkah pengerjaan tugas akhir

## 3.2 Uraian Metodologi

### 3.2.1 Pengambilan Data

Data yang digunakan berupa artikel wisata berbahasa Inggris yang diperoleh dari internet menggunakan teknik *web scraping*. Proses *scraping* dilakukan dengan menggunakan modul *Newspaper* pada *Python* yang memiliki fungsi utama untuk mengekstrak dan mengurai teks dari artikel pada suatu situs web. Proses tersebut dilakukan satu per satu pada setiap artikel dan disimpan dalam sebuah file berekstensi .txt. Data yang didapatkan akan digunakan sebagai data

latih untuk pemodelan dan juga data uji untuk menguji kinerja model yang sudah dibuat. Hasil dari *web scraping* ditunjukkan pada Gambar 3.2.


articles.txt

```
I’ve finally made it! After all these years with a strong passion for Italy and its lovely little towns, I’ve managed to finally go and see the crown gems,

Since getting there is a journey and we landed in Milan, we decided to spend a night in another great city on the Italian coastline, heavily underrated in m

This post contains affiliate links. This means if you click on the link and purchase the item, I will receive an affiliate commission at no additional cost

Thus here are my tips on what to see and do in one day in Genoa, Italy.

I’ve been here before, some 7 years ago, and spent a few days but somehow I didn’t really have any particularly strong memories about this harbor town. So m

Well, apart from the amazing food (best fritto misto I’ve ever had, focaccia Genovese, pesto Genovese, and the list could go on and on), beautiful little to

But after spending one night and discovering the town, I can fully say Genoa is worth visiting!

Read all the things worth knowing when traveling to Italy for the first time.

And it is easy to do in a short time - maybe if you choose to do a cruise on the Mediterranean, you will end up in Genoa as well, thus the question remains:

Of course, one can choose to spend a longer time here and take a day trip from Genoa to Cinque Terre, or a day trip from Genoa to Portofino and other places

But we did it the other way around this time, one day in Genoa on our way to the Cinque Terre.

So here is what to do and see in one day in Genoa, Italy!

Getting to Genoa, Italy

Making your way to Genoa is easy no matter where you are coming from because you will have a lot of great train connections.
```

Gambar 3.2 Hasil *web scraping*

### 3.2.2 *Preprocessing*

*Preprocessing* dilakukan agar teks masukan menjadi lebih mudah dimengerti komputer. Proses *preprocessing* yang akan dilakukan pada penelitian ini adalah penghapusan URL, penghapusan emoji, serta tokenisasi. Proses *stemming* tidak dilakukan karena setelah dicoba ternyata membuat proses pembelajaran menjadi kurang optimal dibandingkan dengan menggunakan data yang tidak melalui proses *stemming*. Proses *preprocessing* berupa *stop words removal* juga tidak dilakukan. Sejatinya *stop words* yang dihapus dapat disesuaikan dengan keinginan dan kebutuhan, namun pemilihan kata-katanya harus dilakukan secara hati-hati. Jika dalam pemilihannya tidak hati-hati, maka dapat berpotensi menghilangkan informasi penting mengenai konteks suatu kata dalam kalimat dan menambah ambiguitas pada suatu kata atau istilah. Misalnya pada kalimat "*You can find various preserved animals in Rahmat International Wildlife Museum & Gallery*", jika kata "*in*" dihapus maka "*Rahmat International Wildlife Museum & Gallery"* yang seharusnya dikenali sebagai tempat wisata, dapat dimaknai secara berbeda. Maka untuk menghindari hal-hal tersebut, penelitian ini tidak menggunakan *stop words removal* pada proses *preprocessing*.

a. Menghapus URL

Penghapusan URL dilakukan karena dalam kasus ini URL tidak memiliki banyak pengaruh. Contoh penghapusan URL ditunjukkan pada Tabel 3.1.

Tabel 3.1 Contoh penghapusan URL

| Sebelum | Sesudah |
|---|---|
| if you love flowers as much as I do, don't shy away from paying a visit to Frida's, the most beautiful flower shop in Bologna. https://www.instagram.com/p/Bu_5CwulG1m/ | if you love flowers as much as I do, don't shy away from paying a visit to Frida's, the most beautiful flower shop in Bologna. |

b. Menghapus *emoji*

Penggunaan *emoji* dalam sebuah artikel *blog* tidak jarang ditemukan. Namun, pada penelitian ini, *emoji* tidak memiliki pengaruh besar terhadap pengenalan nama suatu entitas. Contoh penghapusan *emoji* ditunjukkan pada Tabel 3.2.

Tabel 3.2 Contoh penghapusan *emoji*

| Sebelum | Sesudah |
|---|---|
| We spent most of the time here 😀 | We spent most of the time here |

c. Tokenisasi

Tokenisasi merupakan proses membagi dokumen teks menjadi unit yang lebih kecil, seperti kalimat atau kata. Masing-masing unit yang lebih kecil tersebut disebut dengan istilah token. Tabel 3.3 merupakan contoh dari penerapan tokenisasi pada sebuah kalimat menjadi token kata.

Tabel 3.3 Contoh tokenisasi

| Sebelum | Sesudah |
|---|---|
| The building is both a public library and a museum. | ['The', 'building', 'is', 'both', 'a', 'public', 'library', 'and', 'a', 'museum', '.'] |

### 3.2.3 Pelabelan Data

Pelabelan yang diperlukan adalah pelabelan tempat wisata ke dalam kategorinya masing-masing dan dilakukan secara manual. Terdapat 4 kategori tempat wisata yang dikemukakan oleh (Swarbrooke, 2002), yaitu *natural, man-made purpose-built, man-made non tourist purpose-built,* dan *special events*. Yang termasuk pada kategori pertama adalah tempat

wisata berupa hasil alam seperti pantai, gunung, dan hutan. Yang tergolong kategori kedua adalah tempat wisata yang dibangun secara khusus untuk menarik wisatawan seperti taman, museum, dan galeri. Kategori ketiga berisi tempat wisata yang tujuan pembangunannya bukan untuk menarik wisatawan, contohnya adalah bangunan atau tempat bersejarah seperti monumen, candi, dan kastil. Katedral dan gereja yang merupakan tempat ibadah juga termasuk ke dalam kategori ketiga. Sementara yang tergolong pada kategori terakhir adalah festival dan acara-acara lain seperti acara olahraga.

Beberapa sumber lain juga membagi tempat wisata ke dalam 4 kategori yang diberi nama *natural, purpose, heritage,* dan *event* ("Attractions. - The World of Travel and Tourism.," n.d.; "Types of Attractions – Great Zim Traveller," 2016; "Visitor Attractions - Travel and Tourism Industry," n.d.). Perbedaan dengan pengkategorian yang sebelumnya adalah perbedaan nama pada kategori *man-made non tourist purpose-built* menjadi *heritage*. Walaupun namanya berbeda, namun tempat wisata yang tergolong ke dalamnya tetap sama.

Pada penelitian ini, kategori yang digunakan untuk tempat wisata hanya *natural attraction, heritage attraction,* dan *purposeful built (man-made) attraction*. Kategori *event* tidak dipakai karena dalam pembuatan *dataset* sangat jarang ditemukan. Bagi kata yang bukan merupakan tempat wisata akan masuk ke dalam kategori *outside.* Label yang digunakan ditunjukkan pada Tabel 3.4. Berikut penjelasan masing-masing kategori:

a. *Natural Attraction*

   Tempat wisata yang tergolong ke dalam kategori ini adalah yang merupakan hasil alam dan terbuka untuk umum serta memiliki fasilitas untuk digunakan oleh pengunjung. seperti air terjun, gunung, gua, sungai, gletser, gurun, hutan, danau dan pantai.

b. *Heritage Attraction*

   Kategori ini berisi tempat wisata yang sudah ada sejak lama, kuno, bersejarah, dan seringkali bersifat budaya atau cenderung mewakili budaya dan warisan, serta tempat ibadah. Contohnya adalah reruntuhan, monumen, kuil, benteng, kastil, masjid, gereja, dan katedral.

c. *Purposeful Built (Man-Made) Attraction*

   Tempat wisata yang termasuk kategori ini adalah yang merupakan buatan manusia dan sengaja dibangun untuk menarik pengunjung, seperti bendungan, taman nasional, museum, pasar, taman hiburan, dan galeri.

d. *Outside*

Kata yang bukan merupakan tempat wisata akan tergolong sebagai kategori *outside*, contohnya adalah "*went*", "*good"*, dan "*yesterday*". Kategori *Outside* akan diberi label "O".

Tabel 3.4 Daftar label dan penjelasannya

| Label | Penjelasan | Contoh |
|---|---|---|
| B-NATURAL | Kata pertama (*beginning*) dari nama tempat wisata yang tergolong *natural attraction*. | Franz Josef Glacier:<br>Franz/B-NATURAL<br>Josef/I-NATURAL<br>Glacier/I-NATURAL |
| I-NATURAL | Kata kedua dan seterusnya (*inside*) dari nama tempat wisata yang tergolong *natural attraction*. | |
| B-HERITAGE | Kata pertama (*beginning*) dari nama tempat wisata yang tergolong *heritage attraction*. | Sri Veeramakaliamman Temple:<br>Sri/B-HERITAGE<br>Veeramakaliamman/I-HERITAGE<br>Temple/I-HERITAGE |
| I-HERITAGE | Kata kedua dan seterusnya (*inside*) dari nama tempat wisata yang tergolong *heritage attraction*. | |
| B-PURPOSE | Kata pertama (*beginning*) dari nama tempat wisata yang tergolong *purposeful built (man-made) attraction*. | Gwanghwamun Plaza:<br>Gwanghwamun/B-PURPOSE<br>Plaza/I-PURPOSE |
| I-PURPOSE | Kata kedua dan seterusnya (*inside*) dari nama tempat wisata yang tergolong *purposeful built (man-made) attraction*. | |
| O | Label untuk kata-kata yang bukan merupakan nama tempat wisata (*outside*). | Normal day:<br>Normal/O<br>day/O |

### 3.2.4 Ekstraksi Fitur

Fitur yang digunakan pada penelitian ini, yaitu *Word2Vec*. Tabel 3.5 menunjukkan contoh *Word2Vec* pada beberapa kata dengan vektor berdimensi 5. Angka-angka yang ada di dalam kolom dimensi vektor mewakili bobot dari suatu kata yang didistribusikan di seluruh dimensi. Secara sederhana, setiap dimensi mewakili sebuah makna, dan bobot numerik kata pada dimensi tersebut akan menangkap kedekatan hubungannya dengan makna yang diwakilkan. Namun, makna seperti apa yang diwakilkan pada setiap dimensinya tidak bisa didefinisikan dengan jelas oleh manusia.

Tabel 3.5 Contoh *Word2Vec*

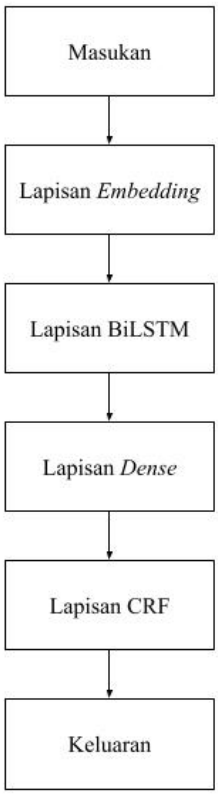
| Kata | Dimensi Vektor | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 |
| Dubai | -1.7890934 | -0.266844 | -0.90740347 | -0.6356304 | 2.1717896 |
| India | -0.97110677 | 0.17386182 | -1.1734697 | -0.46842143 | 1.649491 |
| Indonesia | -0.8502394 | -0.00736618 | -0.9879363 | -0.00659873 | 1.930282 |
| *mountain* | 0.21130832 | 0.8242862 | -1.4168341 | 0.91578346 | 1.2682737 |
| *cliff* | 0.21541822 | 0.5778215 | -1.3274524 | 1.1553676 | 1.3777317 |
| *hill* | 0.18343832 | 0.64653075 | -1.4424297 | 0.8871017 | 1.1769794 |

### 3.2.5 Pemodelan Named Entity Recognition

Langkah awal sebelum membangun model adalah melakukan *splitting data* menjadi data latih dan data uji. Nantinya data latih akan diambil sebagian sebagai data *validation*. Data *validation* adalah data yang digunakan pada proses validasi model guna mencegah terjadinya *overfitting*. *Overfitting* merupakan keadaan ketika model mempelajari detail pada data latih terlalu baik sehingga memiliki tingkat akurasi yang sangat tinggi, tetapi memiliki akurasi yang rendah jika memproses suatu data baru. Data latih lainnya yang tidak masuk ke dalam data *validation* akan digunakan saat melatih model. Setelah model selesai dilatih dan divalidasi, selanjutnya akan diuji menggunakan data uji untuk mengetahui performa dari model yang sudah dibuat sebelumnya.

Langkah selanjutnya, yaitu membangun model. Model ini dibangun dengan menggunakan gabungan dari metode BiLSTM dan CRF seperti yang ditunjukkan pada Gambar 3.3. Data berupa teks akan ditransformasi menjadi vektor menggunakan *Word2Vec* yang selanjutnya akan dilatih dan menghasilkan sebuah kamus. Banyak kata pada kamus tersebut sama dengan jumlah kata unik yang ada pada *dataset* dan setiap kata memiliki vektornya

masing-masing. Selanjutnya, kamus yang berisi vektor yang sudah dilatih sebelumnya akan digunakan sebagai bobot pada lapisan *embedding*. Lapisan *embedding* akan memetakan setiap masukan yang berbentuk *integer* yang mewakili setiap kata ke vektor yang merepresentasikan kata tersebut sesuai dengan yang ada pada kamus hasil pelatihan *Word2Vec.* Vektor hasil *embedding* akan memasuki lapisan BiLSTM yang terdiri dari dua LSTM, yaitu yaitu *forward* LSTM dan *backward* LSTM. *Forward* LSTM akan berjalan dari awal kalimat hingga akhir kalimat, sementara *backward* LSTM berjalan secara terbalik dari akhir ke awal kalimat. Penggabungan dua LSTM tersebut dapat mempertahankan informasi dari masa lalu dan masa depan. Keluaran dari lapisan BiLSTM adalah matriks representasi konteks dari suatu masukan. Selanjutnya matriks tersebut akan diproyeksikan ke lapisan *dense* yang berukuran sebesar 7, yaitu banyaknya jumlah label yang berbeda pada dataset. Lapisan tersebut akan menghasilkan sebuah matriks yang isinya merepresentasikan skor dari setiap token masukan untuk setiap label. Matriks skor tersebut selanjutnya akan masuk ke lapisan CRF.




Gambar 3.3 Alur kerja model BiLSTM-CRF

CRF digunakan sebagai fungsi keputusan untuk menghasilkan urutan label dengan mempertimbangkan label yang ada di sekitarnya sehingga label masa lalu dan masa depan dapat membantu memprediksi label untuk kata saat ini. Hal tersebut bertujuan untuk

menghindari tidak adanya ketergantungan antar label. Terdapat dua jenis skor dalam perhitungan pada CRF, yaitu *emission score* dan *transition score*. *Emission score* pada model ini berasal dari matriks skor keluaran lapisan *dense* sebelumnya. Sedangkan *transition score* awalnya diberi nilai secara acak, nilai tersebut akan diperbarui pada proses pelatihan. Kedua skor tersebut akan digunakan dalam memprediksi urutan label yang akan dijadikan sebagian keluaran suatu kalimat masukan. Urutan label dengan skor prediksi tertinggi akan dipilih sebagai jawaban akhir.

Sebagai contoh, misalnya terdapat kalimat "*We go to Prambanan Temple*" yang dijadikan sebagai masukan pada model BiLSTM-CRF. Pada tahap awal, semua kata pada kalimat tersebut akan diubah menjadi angka, begitu juga dengan label yang akan diubah menjadi angka yang merepresentasikan setiap labelnya. Label yang berbentuk angka tersebut kemudian diubah menjadi matriks yang memiliki nilai biner dan memiliki kolom yang sama dengan jumlah label yang ada, dalam contoh dan penelitian ini terdapat 7 label. Matriks tersebut dapat dilihat pada Tabel 3.6. Sedangkan angka yang merepresentasikan setiap kata pada masukan dapat dilihan pada Tabel 3.7 di bagian masukan untuk lapisan *embedding*. Setiap angka mewakili setiap kata pada kalimat masukan secara berurutan. Tabel tersebut juga menunjukkan masukan dan keluaran lainnya pada setiap lapisan. Pada contoh ini lapisan *embedding* berdimensi 5, sehingga keluaran pada lapisan embedding terdiri dari 5 matriks yang masing-masing merepresentasikan setiap kata pada kalimat masukan secara berurutan, serta setiap matriks terdiri dari 5 kolom yang merupakan ukuran dimensi dari lapisan *embedding* tersebut. Keluaran dari lapisan *embedding* akan masuk ke lapisan BiLSTM. Pada contoh ini setiap LSTM memiliki 5 unit, karena BiLSTM terdiri dari 2 LSTM, maka penggabungan kedua LSTM tersebut menghasilkan 10 unit. Sehingga pada keluaran BiLSTM, setiap matriks memiliki 10 kolom. Setelah itu, keluaran lapisan BiLSTM masuk ke lapisan *dense*. Pada contoh ini, lapisan *dense* memiliki unit sebanyak 7, sehingga pada keluarannya setiap matriks terdiri dari 7 kolom. Keluaran lapisan *dense* akan menjadi masukan pada lapisan terakhir, yaitu CRF. Lapisan CRF menghasilkan prediksi label untuk setiap kata yang masih dalam bentuk matriks representasi label. Jika diterjemahkan menjadi nama label yang sesungguhnya seperti pada Tabel 3.6, kata *"we", "go",* dan *"to"* diberi label O, sedangkan kata "Prambanan" diberi label B-HERITAGE, dan kata *"Temple"* diberi label "I-HERITAGE".

Tabel 3.6 Contoh matriks representasi setiap label

| Nama Label | Matriks Representasi Label |
|---|---|
| B-HERITAGE | [1., 0., 0., 0., 0., 0., 0.] |
| B-NATURAL | [0., 1., 0., 0., 0., 0., 0.] |
| B-PURPOSE | [0., 0., 1., 0., 0., 0., 0.] |
| I-HERITAGE | [0., 0., 0., 1., 0., 0., 0.] |
| I-NATURAL | [0., 0., 0., 0., 1., 0., 0.] |
| I-PURPOSE | [0., 0., 0., 0., 0., 1., 0.] |
| O | [0., 0., 0., 0., 0., 0., 1.] |

Tabel 3.7 Contoh masukan dan keluaran setiap lapisan pada model BiLSTM-CRF

| | | |
|---|---|---|
| Lapisan *Embedding* | Masukan 1x1x5 | array([[ 3901, 16876, 16249, 1990, 7138]], dtype=int32) |
| | Keluaran 1x5x5 | array([<br>[ 1.3276532, -0.1464341, -0.37407014, 0.41752386, 1.2239729],<br>[ 0.89682853, -0.702577, -0.21267204, -0.06746441, 0.5259816],<br>[ 0.61887574, -0.6120933, -0.08085755, -0.06363653, 0.38784117],<br>[ 0.6875345, -0.50314015, -0.14201486, -0.02869875, 0.3741704],<br>[ 0.717283, -1.1027995, -0.61405027, -0.6747531, 0.98590904]],<br>dtype=float32) |
| Lapisan BiLSTM | Masukan 1x5x5 | array([<br>[ 1.3276532, -0.1464341, -0.37407014, 0.41752386, 1.2239729],<br>[ 0.89682853, -0.702577, -0.21267204, -0.06746441, 0.5259816],<br>[ 0.61887574, -0.6120933, -0.08085755, -0.06363653, 0.38784117],<br>[ 0.6875345, -0.50314015, -0.14201486, -0.02869875, 0.3741704],<br>[ 0.717283, -1.1027995, -0.61405027, -0.6747531, 0.98590904]],<br>dtype=float32) |

| | | |
|---|---|---|
| | Keluaran 1x5x10 | array([<br>[-0.04340493, 0.21262889, 0.3158288, 0.17150243, -0.3108446,<br>0.370069, -0.571622, -0.6518946, 0.27461246, 0.3624136],<br>[-0.06982226, 0.37881023, 0.34340054, 0.2647733, -0.3821195,<br>0.34350717, -0.55790424, -0.5674492, 0.17079215, 0.5001609],<br>[-0.11087727, 0.30672964, 0.39520618, 0.3145956, -0.36393526,<br>0.37481508, -0.54982215, -0.4969846, 0.13359112, 0.40452412],<br>[-0.13696606, 0.2726359, 0.44158828, 0.3674969, -0.41144496,<br>0.4579747, -0.55793834, -0.47148234, 0.1383643, 0.40291858],<br>[-0.07940023, 0.36211225, 0.42511573, 0.36132136, -0.568085,<br>0.58652574, -0.5655873, -0.54251635, 0.16316469, 0.39328152]],<br>dtype=float32) |
| Lapisan *Dense* | Masukan 1x5x10 | array([<br>[-0.04340493, 0.21262889, 0.3158288, 0.17150243, -0.3108446,<br>0.370069, -0.571622, -0.6518946, 0.27461246, 0.3624136],<br>[-0.06982226, 0.37881023, 0.34340054, 0.2647733, -0.3821195,<br>0.34350717, -0.55790424, -0.5674492, 0.17079215, 0.5001609],<br>[-0.11087727, 0.30672964, 0.39520618, 0.3145956, -0.36393526,<br>0.37481508, -0.54982215, -0.4969846, 0.13359112, 0.40452412],<br>[-0.13696606, 0.2726359, 0.44158828, 0.3674969, -0.41144496,<br>0.4579747, -0.55793834, -0.47148234, 0.1383643, 0.40291858],<br>[-0.07940023, 0.36211225, 0.42511573, 0.36132136, -0.568085,<br>0.58652574, -0.5655873, -0.54251635, 0.16316469, 0.39328152]],<br>dtype=float32) |
| | Keluaran 1x5x7 | array([<br>[1.0469925, 0.9093942, -0.88977206, -0.06769526, 0.1744933,<br>0.8517888, 0.805452],<br>[ 1.272043, 0.85639805, -0.78275114, 0.13824782, 0.08340643,<br>0.84608686, 0.76893353],<br>[ 1.1380492, 0.7820342, -0.7742618, 0.04618785, 0.10776685,<br>0.7828933, 0.93666637],<br>[ 1.1436803, 0.7634502, -0.8005348, 0.04474042, 0.10707615,<br>0.78251934, 1.073108],<br>[ 1.2718363, 0.75160724, -0.85742843, 0.33700466, -0.1742781,<br>1.0336827, 0.9523978 ]], dtype=float32) |
| Lapisan CRF | Masukan 1x5x7 | array([<br>[1.0469925, 0.9093942, -0.88977206, -0.06769526, 0.1744933,<br>0.8517888, 0.805452],<br>[ 1.272043, 0.85639805, -0.78275114, 0.13824782, 0.08340643,<br>0.84608686, 0.76893353],<br>[ 1.1380492, 0.7820342, -0.7742618, 0.04618785, 0.10776685,<br>0.7828933, 0.93666637],<br>[ 1.1436803, 0.7634502, -0.8005348, 0.04474042, 0.10707615,<br>0.78251934, 1.073108],<br>[ 1.2718363, 0.75160724, -0.85742843, 0.33700466, -0.1742781,<br>1.0336827, 0.9523978 ]], dtype=float32) |

| | | |
|---|---|---|
| | Keluaran 1x5x7 | array([<br>[0., 0., 0., 0., 0., 0., 1.],<br>[0., 0., 0., 0., 0., 0., 1.],<br>[0., 0., 0., 0., 0., 0., 1.],<br>[1., 0., 0., 0., 0., 0., 0.],<br>[0., 0., 0., 1., 0., 0., 0.]], dtype=float32) |

Beberapa skenario dibuat untuk mendapatkan model dengan performa yang terbaik dalam melakukan NER pada domain wisata. Skenario awal akan menggunakan algoritma *Word2Vec* berupa *Skip-gram* karena (Sarkar, 2018) berpendapat bahwa algoritma *Skip-gram* sering kali memberikan hasil yang lebih baik dibandingkan dengan CBOW. Jumlah unit LSTM sebanyak 128 seperti yang digunakan (Wang et al., 2019) dengan fungsi aktivasi *TanH* yang merupakan *default* fungsi aktivasi pada lapisan tersebut. Nilai *dropout* pada lapisan LSTM sebesar 0,5 untuk mencegah terjadinya *overfitting* seperti yang digunakan pada (Sachan, Xie, Sachan, & Xing, 2018; Shen, Yun, Lipton, Kronrod, & Anandkumar, 2018; Wibisono & Khodra, 2018). Jumlah unit pada lapisan *dense* sebanyak 7 yang merupakan total jenis label dengan fungsi aktivasi *TanH*. Fungsi optimasi berupa *Adam* yang biasanya memberikan performa paling baik di antara fungsi optimasi lainnya (Reimers & Gurevych, 2017). Ukuran *batch* 32 dengan *epoch* sebanyak 30 sebagai permulaan dan nilai tersebut akan diganti dengan yang lebih tinggi pada skenario lain.

Terdapat 7 skenario yang akan diuji pada penelitian ini seperti yang terdapat pada Tabel 3.8. Skenario I bertujuan untuk melihat *learning rate* terbaik pada penelitian ini, *learning rate* yang akan diuji coba ada 3, yaitu 0,01, 0,001, dan 0,0001. Skenario II bertujuan untuk melihat algoritma *Word2Vec* mana yang menghasilkan performa terbaik, sehingga terdapat 2 model, yaitu model dengan algoritma *Skip-gram* dan CBOW. Skenario III untuk menguji fungsi aktivasi pada lapisan *dense* dengan menggunakan 3 jenis fungsi aktivasi, yaitu *TanH, ReLU,* dan *linear*. Skenario IV menggunakan 2 ukuran *batch*, yaitu 32 dan 64. Setelah mendapat ukuran *batch* dengan performa yang lebih baik, skenario V dilakukan dengan menggunakan unit LSTM sebanyak 100 dan 128. Lalu skenario VI akan mencoba 2 nilai *epoch*, yaitu pada nilai 30 dan 50. Yang terakhir adalah skenario VII yang mencoba 2 fungsi optimasi, yaitu *Adam* dan *Nadam*. Skenario I sampai VII dilakukan secara bertahap dan berurutan. Konfigurasi yang menghasilkan performa terbaik pada setiap skenarionya akan digunakan pada skenario-skenario berikutnya.

Tabel 3.8 Rangkuman skenario

| **Skenario** | ***Hyperparameter*** | **Konfigurasi** |
|---|---|---|
| I | *Learning rate* | 0,01 |
| | | 0,001 |
| | | 0,0001 |
| II | Algoritma *Word2Vec* | *Skip-gram* |
| | | CBOW |
| III | Fungsi aktivasi lapisan *dense* | *TanH* |
| | | *ReLU* |
| | | *Linear* |
| IV | Ukuran *batch* | 32 |
| | | 64 |
| V | Unit LSTM | 100 |
| | | 128 |
| VI | *epoch* | 30 |
| | | 50 |
| VII | Fungsi optimasi | *Adam* |
| | | *Nadam* |

### 3.2.6 Evaluasi Model

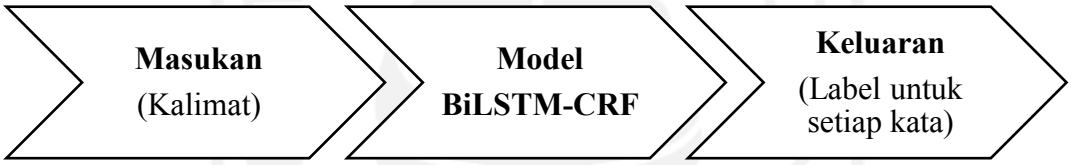
Untuk mengukur kinerja dari NER, (Seok et al., 2016) berpendapat bahwa pengukuran menggunakan F1-*Score* lebih cocok jika dibandingkan dengan akurasi karena sebagian besar label pada data NER merupakan label O, yang mengacu pada token-token yang bukan merupakan sebuah entitas bernama (*named entity)*, dan dengan demikian akurasi tinggi dapat diperoleh. Maka dari itu penelitian ini akan menggunakan F1-*Score* sebagai parameter pengukuran kinerja model. F1-*Score* diperoleh berdasarkan rata-rata harmonik dari *precision* dan *recall* seperti yang ditunjukkan pada Persamaan ( 3.1 ). *Precision* merupakan jumlah data positif yang terklasifikasi dengan benar dibagi dengan jumlah semua data positif. Sedangkan

*recall* merupakan jumlah data positif yang terklasifikasi dengan benar dibagi dengan jumlah semua data yang seharusnya bernilai positif. Nilai terbaik yang dapat dicapai oleh F1-*Score* adalah 1, sedangkan yang terburuk adalah 0. Nilai tersebut juga dapat direpresentasikan dalam bentuk persentase, yaitu dari 0-100% yang juga akan digunakan dalam penelitian ini.

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \quad (3.1)$$

### 3.2.7 Deteksi Entitas

Tahapan ini akan mendeteksi entitas berupa tempat wisata pada suatu masukan. Proses pendeteksian ini menggunakan model yang menghasilkan F1-*Score* terbaik berdasarkan hasil evaluasi skenario sebelumnya. Gambar 3.4 menunjukkan proses pendeteksian entitas.




Gambar 3.4 Proses pendeteksian entitas

# BAB IV
# HASIL DAN PEMBAHASAN

## 4.1 Pengumpulan Data

Data berupa artikel diambil dari beberapa situs web yang ditulis dalam Bahasa Inggris dan membahas tentang wisata di suatu daerah. Proses pencarian artikel dimulai dengan mencari nama kota/negara yang terkenal akan objek-objek wisatanya. Pada awalnya pencarian tersebut dilakukan dengan memasukkan beberapa kata kunci pada mesin pencari *Google* secara bergantian, seperti "*top cities for tourist"*, "*best places to visit*", "*top world heritage sites",* "*world heritage list*", "*best destinations for nature lovers*", dan "*best natural tourist attractions"*. Beberapa hasil yang tertera pada halaman pertama hasil pencarian kemudian dipilih untuk dibaca secara manual. Tidak semua kota/negara yang ditulis pada sumber terkait akan dipilih. Setelah membaca, penulis menentukan daerah mana saja yang selanjutnya akan dicari artikelnya dengan mempertimbangkan variasi wisata yang ada pada daerah tersebut dan daerah lain yang sudah dipilih sebelumnya, agar tempat wisata pada dataset semakin beragam. Setelah memilih beberapa nama kota/negara, proses selanjutnya adalah mencari artikel yang berhubungan dengan kota/negara tersebut. Pencarian dilakukan dengan memasukkan kata kunci "*best tourist attractions in **" simbol * akan diganti dengan nama kota/negara tertentu, misalnya "*best tourist attractions in Sao Paulo*". Mesin pencari akan menampilkan beberapa hasil dari situs web yang berbeda, dan nantinya tautan dari situs web yang dipilih akan disalin untuk di-*scraping* menggunakan modul *Newspaper,* namun tidak semua situs berhasil di-*scraping* menggunakan modul tersebut. Sehingga kemudian pemilihan artikel akan menggunakan dua pertimbangan, yaitu variasi jenis tempat wisata yang tertera dan apabila artikel tersebut berhasil di-*scraping*. Jika 1 artikel yang dipilih sudah berhasil di-*scraping,* maka proses pencarian akan lanjut untuk mencari artikel pada kota/negara lainnya.

Selain tahap pencarian yang di atas, untuk mempercepat pengumpulan artikel yang memuat banyak tempat wisata, dilakukan proses pencarian lain yang kata kuncinya berbeda dengan sebelumnya. Pada tahap ini, pencarian dilakukan dengan memasukkan kata kunci "*best * in the world*". Simbol * akan diganti dengan nama objek yang biasanya dijadikan tempat wisata seperti *caves, lakes, waterfalls, glaciers, craters, castles, ruins, beaches, mountain, fjord, valley, canyon, lagoon, desert, wetlands, rivers, synagogue, church, mausoleum,*

*mosque, forest, monastery, lighthouse, opera house, mall, pyramid, sea cliff, monolith, tower, geyser, canal,* dan *bazaar*. Kata kunci lain seperti "*best lakes in Indonesia*" dan "*best beaches in Indonesia*" juga digunakan. Tahapan berikutnya sama dengan yang sebelumnya sudah dijelaskan. Terdapat juga beberapa artikel yang diambil langsung tanpa proses pencarian melalui *Google* dari situs web *touropia.com* dan *traveltriangle.com* yang sebelumnya sudah ada artikel yang berhasil di-*scraping* dari kedua situs web tersebut. Proses pengumpulan data berhenti ketika semua kategori sudah berjumlah minimal 1.400.

Pada akhirnya, artikel yang berhasil dikumpulkan berjumlah sebanyak 92 dengan jumlah kata unik sebanyak 17.137. Artikel-artikel tersebut diperoleh dari beberapa situs web, yaitu *worldofwanderlust.com, touropia.com, tourmyindia.com, vaticantour.com, myguidezimbabwe.com, daytours4u.com, traveloka.com, traveltips.usatoday.com, traveltriangle.com, tripsavvy.com, wefindyougo.com, eatingeurope.com, youngadventuress.com, wowtravel.me, we12travel.com, greenglobaltravel.com, dreamworldtravel.co.uk, signatureluxurytravel.com, thevoyageurscollection.co.uk, allindonesiatourism.com, heyalma.com, hiconsumption.com, phenomenalplace.com, traveller.co.au.*

## 4.2 *Preprocessing*

Setelah data berhasil dikumpulkan, tahap selanjutnya adalah melakukan *preprocessing*, proses *preprocessing* yang dilakukan sebagai berikut:

a. Menghapus URL

Pada Gambar 4.1 menunjukkan implementasi kode program untuk menghapus URL.

```
def remove_url(text):
  text = re.sub(r'(?i)\b((?:https?://|www\d{0,3}[.]|[a-z0-9.\-]+[.][a-
z]{2,4}/)(?:[^\s()<>]+|\(([^\s()<>]+|(\([^\s()<>]+\)))*\))+(?:\(([^\s()<>]+|(\([^\s
()<>]+\)))*\)|[^\s`!()\[\]{};:\'".,<>?«»“”‘’]))', '', text)
  return text
```

Gambar 4.1 Kode program menghapus URL

b. Menghapus *emoji*

Pada Gambar 4.2 menunjukkan implementasi kode program untuk menghapus *emoji*.

```
def remove_emoji(text):
  regex_pattern = re.compile(pattern = "["
        u"\U0001F600-\U0001F64F"  # emoticons
        u"\U0001F300-\U0001F5FF"  # symbols & pictographs
        u"\U0001F680-\U0001F6FF"  # transport & map symbols
        u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                           "]+", flags = re.UNICODE)
  return regex_pattern.sub(r'.',text)
```

Gambar 4.2 Kode program menghapus *emoji*

c. Tokenisasi

Gambar 4.3 menunjukkan implementasi kode program untuk melakukan tokenisasi. Proses tokenisasi diawali dengan memecah teks yang berbentuk paragraf menjadi token kalimat. Setiap kalimat akan diberi nomor kalimat. Setelah itu token kalimat tersebut dipecah menjadi token kata yang selanjutnya disimpan dengan format csv beserta nomor kalimatnya. Tahap ini menghasilkan sebanyak 183.507 token.

```
def sentTokenize(text):
  text = nltk.sent_tokenize(text)
  return text

def wordTokenize(text):
  text = nltk.word_tokenize(text)
  return text
```

Gambar 4.3 Kode program tokenisasi

## 4.3 Pelabelan Data

Hasil dari *preprocessing* berupa data yang sudah dipecah menjadi token selanjutnya akan diberi label sesuai dengan kategorinya seperti yang ditunjukkan pada Gambar 4.4. Sedangkan untuk melihat jumlah token pada setiap label dapat dilihat pada Tabel 4.1. *Dataset* ini sangat tidak seimbang karena jumlah label O jauh lebih banyak dibandingkan dengan label lain.

| | A | B | C |
|---|---|---|---|
| 115267 | | . | O |
| 115268 | Sentence 5247 | Great | B-NATURAL |
| 115269 | | Blue | I-NATURAL |
| 115270 | | Hole | I-NATURAL |
| 115271 | | The | O |
| 115272 | | most | O |
| 115273 | | popular | O |
| 115274 | | dive | O |
| 115275 | | destination | O |
| 115276 | | in | O |
| 115277 | | Belize | O |
| 115278 | | , | O |
| 115279 | | the | O |
| 115280 | | Great | B-NATURAL |
| 115281 | | Blue | I-NATURAL |
| 115282 | | Hole | I-NATURAL |
| 115283 | | offers | O |
| 115284 | | divers | O |
| 115285 | | interesting | O |
| 115286 | | observations | O |

Gambar 4.4 Hasil pelabelan data

Tabel 4.1 Jumlah token per label

| Label | Jumlah Token |
|---|---|
| O | 171.728 |
| B-NATURAL | 1.789 |
| I-NATURAL | 1.853 |
| B-HERITAGE | 1.401 |
| I-HERITAGE | 2.051 |
| B-PURPOSE | 1.811 |
| I-PURPOSE | 2.874 |

## 4.4 Persiapan Data Masukan

Terdapat dua proses persiapan data masukan yang dilakukan, yaitu untuk masukan model BiLSTM-CRF dan masukan pada kelas *Word2Vec*. Berikut penjelasan kedua proses tersebut:

**Data masukan untuk model BiLSTM-CRF**

Sebelum digunakan sebagai data masukan, data pada *dataset* dibentuk terlebih dahulu menjadi sebuah *list* yang di dalamnya berisi *list* yang berisikan *tuple* menggunakan kelas *SentenceGetter*. Hal tersebut bertujuan untuk membedakan antara satu kalimat dengan yang lainnya. Kalimat yang berada di dalam list tersebut selanjutnya akan disebut sebagai sampel. Total sampel pada penelitian ini adalah 8.500. Selanjutnya setiap kata dan label akan diubah menjadi indeks angka yang mewakili setiap kata dan label. Setiap sampel harus memiliki panjang yang sama dan penelitian ini menggunakan sampel dengan panjang 90. Fungsi *pad_sequences()* digunakan untuk memastikan bahwa setiap sampel mempunyai panjang yang sama dengan cara memotong sampel yang memiliki panjang lebih dari 90 dan menambah sampel yang memiliki panjang kurang dari 90 menggunakan angka 0. Kode program pada tahap ini dapat dilihat pada Gambar 4.5.

```
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["Words"].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(data["Words"].values))
n_words = len(words); n_words
tags = sorted(set(data["Tag"].values))
n_tags = len(tags); n_tags


max_len = 90
word2idx = {w: i + 1 for i, w in enumerate(words)}
tag2idx = {t: i for i, t in enumerate(tags)}

from keras.preprocessing.sequence import pad_sequences

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

y = [[tag2idx[w[1]] for w in s] for s in sentences]
y = pad_sequences(maxlen=max_len, sequences=y, padding="post", value=tag2idx["O"])
```

Gambar 4.5 Kode program persiapan data masukan untuk model BiLSTM-CRF

**Data masukan untuk kelas *Word2Vec***

Pada tahap ini *dataset* akan dibentuk terlebih dahulu menjadi sebuah *list* sampel menggunakan kelas *SentenceGetter2*. Hal ini juga bertujuan untuk membedakan antara satu kalimat dengan yang lainnya. Kode program pada tahap ini dapat dilihat pada Gambar 4.6.

```
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["Words"].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

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

Gambar 4.6 Kode program persiapan data masukan untuk kelas *Word2Vec*

## 4.5 Ekstraksi Fitur

Setelah *dataset* terbentuk, langkah selanjutnya adalah melakukan *word embedding* menggunakan teknik *Word2Vec* untuk merepresentasikan kata dalam bentuk vektor yang akan digunakan sebagai bobot pada lapisan *embedding*. Proses ini dilakukan dengan menggunakan *library Gensim* dengan nama kelas *Word2Vec* seperti yang ditunjukkan pada Gambar 4.7. Penelitian ini menggunakan vektor dengan panjang 100 yang merupakan ukuran *default* dari kelas tersebut.

```
model_w2v = gensim.models.Word2Vec(sentences=sentences2, size=100, window=10, min_c
ount=1, sg=1, seed=1)
```

Gambar 4.7 Kode program *Word2Vec*

## 4.6 Pemodelan NER

a. *Splitting Data*

Pembagian data berupa sampel ke dalam data latih dan data uji pada kasus ini menggunakan *library Scikit-Learn* dengan nama kelas *StratifiedKFold*. *StratifiedKFold* akan membagi sampel ke dalam sejumlah K *subset*. Penggunaan *StratifiedKFold* bertujuan untuk membagi

data latih dan data uji dengan perbandingan jumlah kelas yang relatif sama pada setiap *subset*-nya. Inisialisasi nilai K terdapat pada parameter n_splits. Selanjutnya data uji diatur sejumlah 1/K bagian dari keseluruhan sampel. Penelitian ini menggunakan nilai K sebesar 10 sehingga sampel untuk data latih berjumlah 7.650 dan data uji sebanyak 850 sampel. Gambar 4.8 menunjukkan kode program untuk *splitting data*.

```
skf = StratifiedKFold(n_splits=n_split, shuffle=True, random_state=1)

for i, (train, test) in enumerate(skf.split(X,y.argmax(1))):
  X_train, X_test = X[train], X[test]
  y_tr, y_te = y[train], y[test]
```

Gambar 4.8 Kode program *splitting data*

b. Membangun Model BiLSTM-CRF

Model dibangun dengan menggunakan tipe *sequential*. Pada tipe tersebut, lapisan baru ditambahkan dengan menggunakan fungsi '*add*()' secara berurutan. *Activation* merupakan fungsi aktivasi pada lapisan model yang bertujuan untuk memetakan keluaran tertentu menjadi suatu himpunan masukan tertentu. Dengan demikian, fungsi aktivasi memiliki pengaruh terhadap hasil dan akurasi dari model yang dibangun. Terdapat juga lapisan *dense* yang menghubungkan semua *node* dari lapisan sebelumnya ke semua *node* di lapisan terkini. Lapisan *dense* tersebut dibungkus menggunakan kelas *TimesDistributed* yang akan memastikan bahwa lapisan *dense* yang sama akan diterapkan pada setiap *list* data masukan dengan menggunakan set bobot yang sama. Lapisan terakhir adalah lapisan CRF dengan keluaran sejumlah 7 yang menandakan seluruh jenis label.

Langkah selanjutnya, yaitu *compile* model. Ada beberapa parameter yang digunakan, yaitu *optimizer*, *loss*, dan *metrics*. *Optimizer* (fungsi optimasi) adalah metode yang digunakan untuk mengganti atribut-atribut dari *neural network*, seperti *learning rate* dan juga bobot dengan tujuan untuk menurunkan nilai *loss* dan memberikan hasil yang lebih akurat. *Loss* adalah sebuah prediksi eror pada *neural network* yang digunakan dalam perhitungan gradien untuk memperbarui bobot dari *neural network*. *Metrics* adalah sebuah fungsi yang bisa digunakan untuk menilai performa model. *Loss* yang digunakan pada penelitian ini berupa *crf_loss* dan *metrics* berupa *crf_accuracy*. Gambar 4.9 menunjukkan kode program untuk membangun model BiLSTM-CRF.

```
def blstmcrf():

  model = Sequential()
  model.add(embedding_layer)
  model.add(Bidirectional(LSTM(units=128, activation="tanh", return_sequences=True,
                               dropout=0.5)))
  model.add(TimeDistributed(Dense(n_tags, activation="tanh")))
  crf_layer = CRF(n_tags, test_mode='viterbi')
  model.add(crf_layer)

  print(model.summary())
  adam = Adam(lr=0.001)
  model.compile(optimizer=adam, loss=crf_layer.loss_function, metrics=[crf_layer.ac
curacy])

  return model
```

Gambar 4.9 Kode program membangun model BiLSTM-CRF

c. Melatih Model

Model dilatih menggunakan fungsi '*fit*()' dengan 5 parameter, yaitu data masukan (X_train), target data (y_train), *epoch*, *validation split*, dan *batch size*. Jumlah *epoch* menentukan berapa kali data latih akan dilatih pada model. Namun, akan terlalu besar jika seluruh data diproses secara bersamaan pada setiap *epoch*-nya, maka dari itu data dibagi ke dalam beberapa *batch* dengan ukuran yang lebih kecil untuk diproses, hal tersebut dapat diatur pada *batch size*. Parameter *validation split* memisahkan sebagian data latih menjadi data validasi. Nilai *validation split* pada penelitian ini adalah 0.2, yang berarti 20% data latih akan digunakan sebagai data validasi sehingga data latih menjadi sebanyak 6.120 sampel dan data validasi 1.530 sampel. Gambar 4.10 menunjukkan kode program untuk melatih model.

```
history = model.fit(X_train, y_train, epochs=epochs,
                    validation_split=0.2, batch_size=32)
```

Gambar 4.10 Kode program melatih model

## 4.7 Evaluasi

Proses evaluasi menggunakan data uji dan bertujuan untuk mengukur kinerja dari model BiLSTM-CRF yang dibuat terhadap data uji. Performa model pada setiap skenario akan dibandingkan satu sama lain berdasarkan rata-rata F1-*Score* yang diperoleh untuk mencari model terbaik. Hasil evaluasi beserta analisis skenario dijabarkan sebagai berikut:

a. Skenario I

Dari hasil evaluasi yang ditunjukkan pada Tabel 4.2, dapat dilihat bahwa perbedaan *learning rate* memiliki dampak yang cukup signifikan terhadap performa model. Dari tiga nilai *learning rate* yang diuji, performa terbaik berdasarkan rata-rata F1-*Score* dihasilkan model dengan *learning rate* 0,001, lalu yang kedua diperoleh dengan *learning rate* sebesar 0,0001, dan yang terakhir dengan *learning rate* sebesar 0,01. F1-*Score* yang rendah pada model dengan *learning rate* 0,01 disebabkan karena dalam kasus ini, nilai tersebut termasuk besar sehingga dapat menyebabkan nilai bobot meningkat secara drastis dan model menjadi tidak stabil, serta akhirnya menghasilkan performa yang kurang baik. Sementara untuk *learning rate* yang kecil, misalnya dalam kasus ini sebesar 0,0001 akan menyebabkan proses pelatihan menjadi lambat, dan dibutuhkan epoch yang lebih banyak untuk menghasilkan performa yang lebih baik. Maka dari itu, F1-*Score* untuk skenario berikutnya akan tetap menggunakan nilai 0,001.

Tabel 4.2 Hasil evaluasi skenario I

| **Nomor *Fold*** | **F1-*Score*** | | |
|---|---|---|---|
| | ***Learning Rate* 0,01** | ***Learning Rate* 0,001** | ***Learning Rate* 0,0001** |
| 1 | 13,1% | 37,9% | 1% |
| 2 | 19,1% | 44,8% | 25,2% |
| 3 | 12,6% | 53,9% | 32% |
| 4 | 17,8% | 64,9% | 38,8% |
| 5 | 18,1% | 74,6% | 45,1% |
| 6 | 15,5% | 80% | 44,9% |
| 7 | 16,6% | 81,5% | 49,4% |
| 8 | 13,1% | 79,5% | 48,9% |
| 9 | 6% | 84% | 49,3% |
| 10 | 11% | 85,8% | 53,8% |
| **Rata-Rata** | **14,27%** | **68,68%** | **38,83%** |

b. Skenario II

Dari hasil evaluasi yang ditunjukkan pada Tabel 4.3, dapat dilihat bahwa model yang menggunakan algoritma *Word2Vec* berupa *Skip-gram* memiliki hasil yang lebih baik dibandingkan dengan model yang menggunakan algoritma *Word2Vec* berupa CBOW. Penggunaan *Skip-gram* juga menghasilkan F1-*Score* yang lebih tinggi pada setiap *fold*-nya. Perbedaan rata-rata F1-*Score* yang dihasilkan kedua algoritma tersebut sebesar 15,67%. Sehingga untuk kasus ini, skenario selanjutnya akan menggunakan Word2Vec dengan algoritma *Skip-gram*.

Tabel 4.3 Hasil evaluasi skenario II

| **Nomor *Fold*** | **F1-*Score*** | |
|---|---|---|
| | ***Skip-gram*** | **CBOW** |
| 1 | 37,9% | 18,4% |
| 2 | 44,8% | 26,7% |
| 3 | 53,9% | 33,5% |
| 4 | 64,9% | 42,7% |
| 5 | 74,6% | 57,4% |
| 6 | 80% | 58,1% |
| 7 | 81,5% | 75,8% |
| 8 | 79,5% | 69,4% |
| 9 | 84% | 71,9% |
| 10 | 85,8% | 76,3% |
| **Rata-Rata** | **68,68%** | **53,01%** |

c. Skenario III

Hasil evaluasi yang tertera pada Tabel 4.4 menunjukkan bahwa dari tiga fungsi aktivasi berbeda yang digunakan pada lapisan *dense*, fungsi aktivasi *linear* memperoleh rata-rata F1-*Score* tertinggi dibandingkan dengan *TanH* dan *ReLU*, yaitu sebesar 71,66%. Posisi kedua ditempati oleh *TanH* dengan F1-*Score* sebesar 68,68%, dan yang terakhir adalah *ReLU* dengan F1-*Score* sebesar 54,45%. Lapisan *dense* pada skenario selanjutnya akan menggunakan fungsi aktivasi *linear*.

Tabel 4.4 Hasil evaluasi skenario III

| Nomor *Fold* | F1-*Score* | | |
|---|---|---|---|
| | **Fungsi Aktivasi *TanH*** | **Fungsi Aktivasi *Linear*** | **Fungsi Aktivasi *ReLU*** |
| 1 | 37,9% | 47,6% | 27,9% |
| 2 | 44,8% | 51,3% | 33,7% |
| 3 | 53,9% | 56% | 43,2% |
| 4 | 64,9% | 67% | 47% |
| 5 | 74,6% | 76,4% | 59,8% |
| 6 | 80% | 83,5% | 60,2% |
| 7 | 81,5% | 83,2% | 68,4% |
| 8 | 79,5% | 82,9% | 66,30% |
| 9 | 84% | 85% | 67,10% |
| 10 | 85,8% | 83,8% | 70,9% |
| **Rata-Rata** | **68,68%** | **71,66%** | **54,45%** |

d. Skenario IV

Dari hasil evaluasi yang ditunjukkan pada Tabel 4.5, dapat dilihat bahwa model dengan ukuran *batch* 32 memperoleh rata-rata F1-*Score* sebesar 71,66%, sedangkan model dengan ukuran *batch* 64 memperoleh rata-rata F1-*Score* sebesar 67%. Pada kasus ini, ukuran *batch* yang lebih kecil memiliki rata-rata F1-*Score* lebih tinggi daripada model dengan ukuran *batch* yang lebih besar, perbedaan di antara kedua rata-ratanya adalah sebesar 4,66%. Skenario selanjutnya akan menggunakan ukuran *batch* sebesar 32.

Tabel 4.5 Hasil evaluasi skenario IV

| Nomor *Fold* | F1-*Score* | |
|---|---|---|
| | **Ukuran *Batch* 32** | **Ukuran *Batch* 64** |
| 1 | 47,6% | 40,6% |
| 2 | 51,3% | 46,4% |
| 3 | 56% | 47,8% |
| 4 | 67% | 54,8% |
| 5 | 76,4% | 65,8% |
| 6 | 83,5% | 78% |
| 7 | 83,2% | 83,2% |
| 8 | 82,9% | 82,2% |
| 9 | 85% | 86% |
| 10 | 83,8% | 85,1% |
| **Rata-Rata** | **71,66%** | **67%** |

e. Skenario V

Dari hasil evaluasi yang ditunjukkan pada Tabel 4.6, dapat dilihat bahwa model dengan jumlah unit LSTM sebesar 100 memiliki rata-rata F1-*Score* sebesar 66% dan model dengan jumlah unit LSTM sebesar 128 memiliki rata-rata F1-*Score* sebesar 71,66%. Perbedaan antara hasil keduanya sebesar 5,66% dan model dengan unit LSTM yang lebih banyak menghasilkan performa yang lebih baik. Maka untuk skenario selanjutnya akan menggunakan unit LSTM sebanyak 128.

Tabel 4.6 Hasil evaluasi skenario V

| Nomor *Fold* | F1-*Score* | |
|---|---|---|
| | **Unit LSTM 100** | **Unit LSTM 128** |
| 1 | 41,3% | 47,6% |
| 2 | 47,3% | 51,3% |
| 3 | 49,6% | 56% |
| 4 | 57,9% | 67% |
| 5 | 67,6% | 76,4% |
| 6 | 75,7% | 83,5% |
| 7 | 79,3% | 83,2% |
| 8 | 76,5% | 82,9% |
| 9 | 80,5% | 85% |
| 10 | 84,3% | 83,8% |
| **Rata-Rata** | **66%** | **71,66%** |

f. Skenario VI

Dari hasil evaluasi yang ditunjukkan pada Tabel 4.7, dapat dilihat bahwa model dengan *epoch* sebesar 30 menghasilkan rata-rata F1-*Score* sebesar 71,66%, sedangkan model dengan nilai *epoch* sebesar 50 menghasilkan F1-*Score* sebesar 75,25%. Terdapat perbedaan sebesar 3,59% pada kedua hasil tersebut. Karena model dengan nilai *epoch* sebesar 50 memiliki performa yang lebih baik, maka skenario selanjutnya akan menggunakan nilai *epoch* sebesar 50.

Tabel 4.7 Hasil evaluasi skenario VI

| **Nomor *Fold*** | **F1-*Score*** | |
|---|---|---|
| | ***Epoch* 30** | ***Epoch* 50** |
| 1 | 47,6% | 46,7% |
| 2 | 51,3% | 51,6% |
| 3 | 56% | 65,1% |
| 4 | 67% | 77,4% |
| 5 | 76,4% | 84,6% |
| 6 | 83,5% | 86,7% |
| 7 | 83,2% | 85,3% |
| 8 | 82,9% | 83,2% |
| 9 | 85% | 86,7% |
| 10 | 83,8% | 85,5% |
| **Rata-Rata** | **71,66%** | **75,25%** |

g. Skenario VII

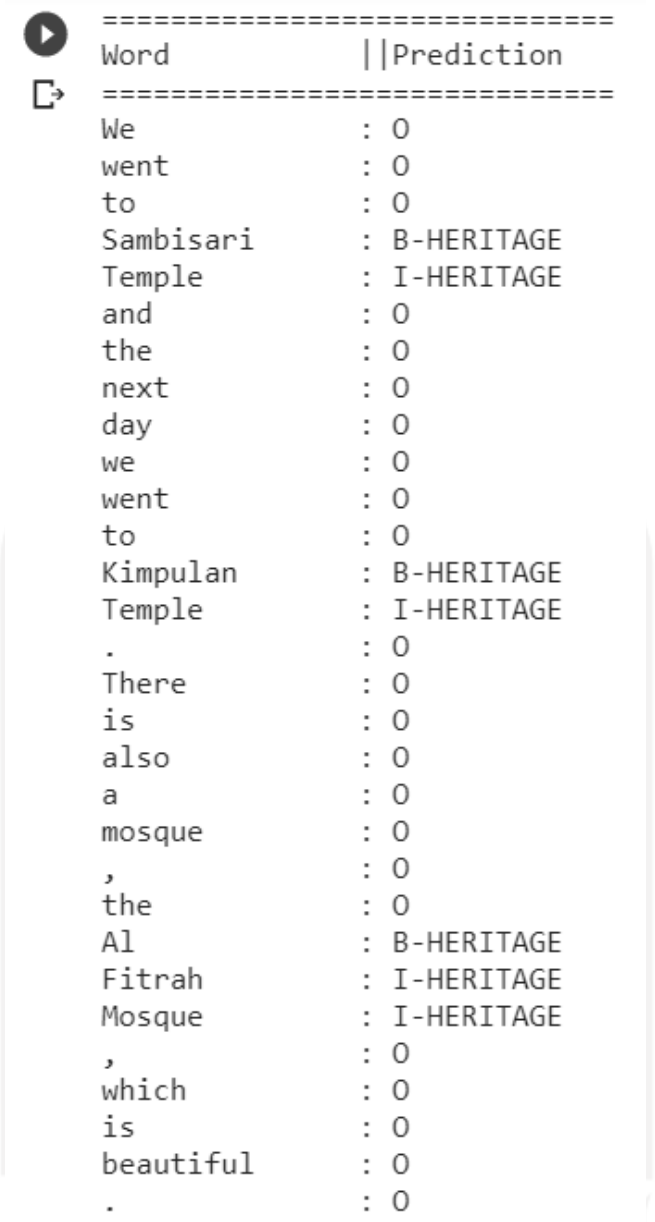
Tabel 4.8 menunjukkan bahwa model yang menggunakan fungsi optimasi *Adam* memperoleh rata-rata F1-*Score* sebesar 75,25% dan model yang menggunakan fungsi optimasi *Nadam* mendapat rata-rata F1-*Score* sebesar 74,61%. Perbedaan hasil antara keduanya sangat sedikit, yaitu sebesar 0,64% dengan performa terbaik diperoleh oleh model dengan fungsi optimasi *Adam*.

Tabel 4.8 Hasil evaluasi skenario VII

| Nomor *Fold* | F1-*Score* | |
|---|---|---|
| | **Fungsi Optimasi *Adam*** | **Fungsi Optimasi *Nadam*** |
| 1 | 46,7% | 44,8% |
| 2 | 51,6% | 51,9% |
| 3 | 65,1% | 63,7% |
| 4 | 77,4% | 76,1% |
| 5 | 84,6% | 82,3% |
| 6 | 86,7% | 84,2% |
| 7 | 85,3% | 86,3% |
| 8 | 83,2% | 84,4% |
| 9 | 86,7% | 84,8% |
| 10 | 85,5% | 87,7% |
| **Rata-Rata** | **75,25%** | **74,61%** |

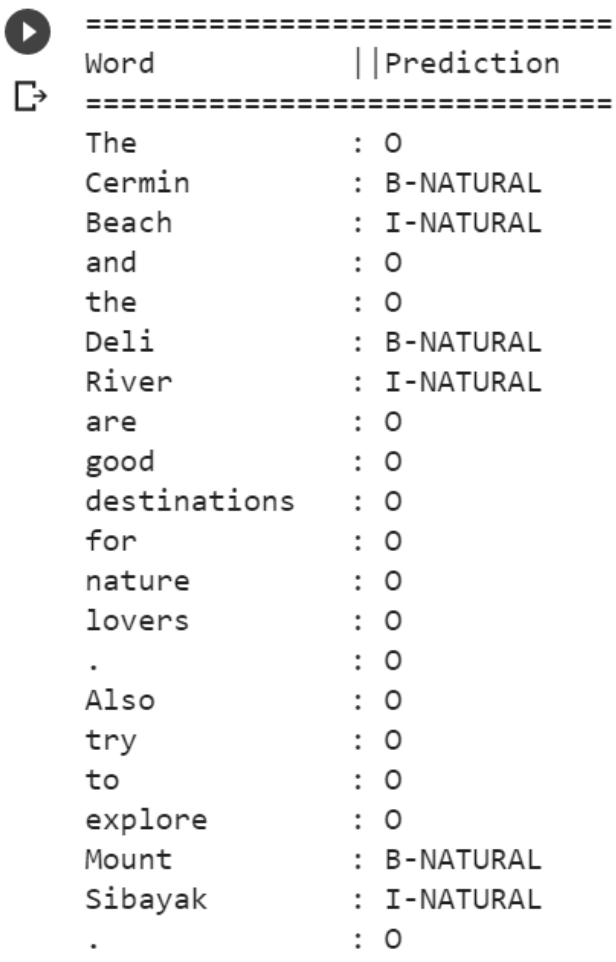
## 4.8 Deteksi Entitas

Model dengan skenario terbaik berdasarkan skenario-skenario yang sudah dilakukan menggunakan *learning rate* sebesar 0,001, algoritma *Word2Vec* berupa *Skip-gram*, fungsi aktivasi pada lapisan *dense* berupa *linear*, ukuran *batch 32*, unit LSTM 128, *epoch* sebanyak 50, dan fungsi optimasi *Adam*. Selanjutnya, skenario terbaik yang telah diperoleh akan diimplementasikan untuk melakukan pendeteksian entitas pada data artikel wisata. Entitas yang dideteksi berupa tempat wisata yang digolongkan ke dalam 3 kategori, yaitu *Heritage Attraction, Purposeful Built (Man-Made) Attraction,* dan *Natural Attraction*. Sedangkan untuk kata yang tidak termasuk tempat wisata akan termasuk ke dalam kategori *outside*. Label untuk setiap kategori sama dengan yang tercantum pada Tabel 3.4. Gambar 4.11, Gambar 4.12, Gambar 4.13, dan Gambar 4.14 adalah hasil pendeteksian entitas menggunakan model dengan rata-rata F1-*Score* terbaik. Gambar 4.11, Gambar 4.12, Gambar 4.13 menggunakan sebuah kalimat masukan yang dibuat oleh penulis dan sebelumnya sudah dipastikan bahwa tempat wisata yang tercantum pada kalimat tersebut tidak terdapat pada *dataset*. Gambar 4.14 menggunakan masukan yang berasal dari artikel mengenai wisata di kota Tokyo yang diambil dari situs web *touropia.com* yang juga tidak terdapat pada *dataset*.



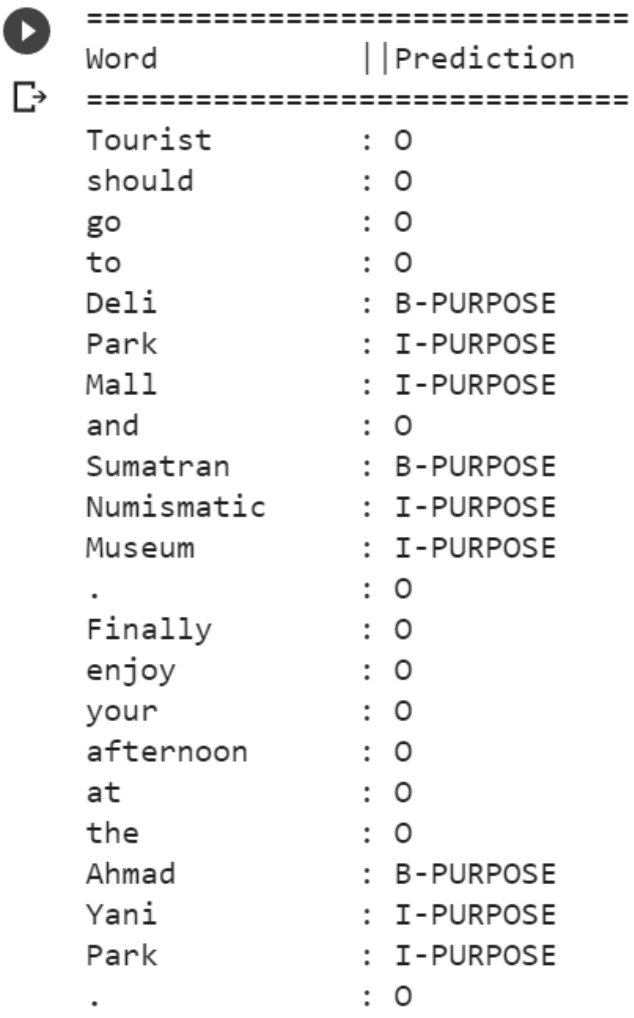
```
==============================
Word            ||Prediction
==============================
We              : O
went            : O
to              : O
Sambisari       : B-HERITAGE
Temple          : I-HERITAGE
and             : O
the             : O
next            : O
day             : O
we              : O
went            : O
to              : O
Kimpulan        : B-HERITAGE
Temple          : I-HERITAGE
.               : O
There           : O
is              : O
also            : O
a               : O
mosque          : O
,               : O
the             : O
Al              : B-HERITAGE
Fitrah          : I-HERITAGE
Mosque          : I-HERITAGE
,               : O
which           : O
is              : O
beautiful       : O
.               : O
```

Gambar 4.11 Contoh hasil deteksi entitas dengan kategori *heritage*



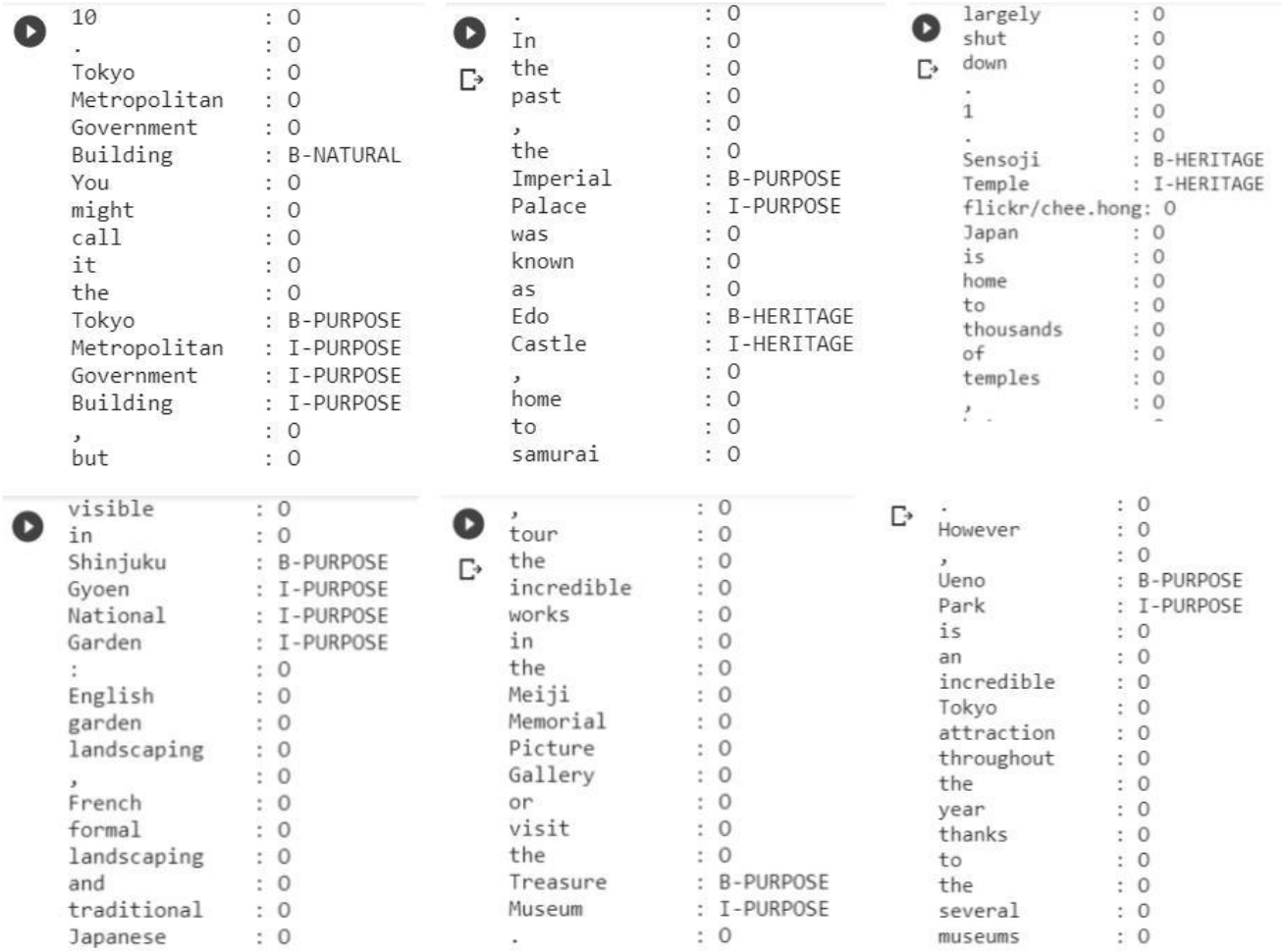
```
==============================
Word            ||Prediction
==============================
The             : O
Cermin          : B-NATURAL
Beach           : I-NATURAL
and             : O
the             : O
Deli            : B-NATURAL
River           : I-NATURAL
are             : O
good            : O
destinations    : O
for             : O
nature          : O
lovers          : O
.               : O
Also            : O
try             : O
to              : O
explore         : O
Mount           : B-NATURAL
Sibayak         : I-NATURAL
.               : O
```

Gambar 4.12 Contoh hasil deteksi entitas dengan kategori *natural*



```
==============================
Word            ||Prediction
==============================
Tourist         : O
should          : O
go              : O
to              : O
Deli            : B-PURPOSE
Park            : I-PURPOSE
Mall            : I-PURPOSE
and             : O
Sumatran        : B-PURPOSE
Numismatic      : I-PURPOSE
Museum          : I-PURPOSE
.               : O
Finally         : O
enjoy           : O
your            : O
afternoon       : O
at              : O
the             : O
Ahmad           : B-PURPOSE
Yani            : I-PURPOSE
Park            : I-PURPOSE
.               : O
```

Gambar 4.13 Contoh hasil deteksi entitas dengan kategori *purpose*




Gambar 4.14 Contoh hasil deteksi entitas pada suatu artikel yang sama

Berdasarkan hasil deteksi entitas yang sudah dilakukan dengan model yang memiliki rata-rata F1-*Score* sebesar 75,25%, terdapat beberapa entitas yang berhasil diprediksi dengan benar walaupun nilai tersebut masih jauh dari sempurna. Seluruh entitas pada Gambar 4.11, Gambar 4.12, dan Gambar 4.13 berhasil diprediksi dengan benar. Ditemukan beberapa kesalahan saat memprediksi entitas suatu kata seperti yang ditunjukkan pada Gambar 4.14, yaitu "*Tokyo Metropolitan Government Building*" yang seharusnya tergolong *purpose*, namun kata "*Tokyo Metropolitan Government*" diprediksi sebagai *outside* dan "*Building*" diprediksi sebagai *natural*. "*Imperial Palace*" yang seharusnya diprediksi sebagai *heritage*, tergolong sebagai *purpose*. Kesalahan juga terdapat pada "*Meiji Memorial Picture Gallery"* yang seharusnya termasuk *purpose*, namun diprediksi sebagai *outside*. Hal tersebut diakibatkan kinerja model yang masih perlu ditingkatkan lagi pada saat melakukan pembelajaran, serta beberapa jenis tempat wisata yang jumlahnya masih sedikit di dalam *dataset* sehingga untuk memprediksinya menjadi lebih sulit.

# BAB V
# KESIMPULAN DAN SARAN

## 5.1 Kesimpulan

Berdasarkan penelitian mengenai *named entity recognition* pada artikel wisata menggunakan gabungan metode BiLSTM dan CRF dapat disimpulkan bahwa:

a. Gabungan metode BiLSTM dan CRF terbukti mampu mendeteksi entitas tempat wisata pada suatu artikel. Pada proses pendeteksiannya, tahapan yang perlu dilakukan, yaitu pengumpulan data, *preprocessing*, pelabelan data, ekstraksi fitur, pemodelan NER, evaluasi model, dan deteksi entitas.
b. Penelitian ini menghasilkan model yang dapat melakukan prediksi dengan cukup baik, namun masih terdapat kesalahan pada pendeteksiannya. Dari seluruh skenario yang telah dilakukan, rata-rata F1-*Score* yang dihasilkan model terbaik sebesar 75,25%.

## 5.2 Saran

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

a. Memperbaiki kualitas *dataset* dan juga menambah datanya dengan tempat wisata yang lebih banyak dan beragam.
b. Menambah entitas yang dapat dikenali pada *dataset*, seperti nama kota, waktu kunjungan, biaya masuk, dan sebagainya.
c. Mencoba membangun versi NER lain seperti *nested* NER.
d. Menggunakan metode lain pada saat membangun model NER.

# DAFTAR PUSTAKA


Alfred, R., Leong, L. C., On, C. K., & Anthony, P. (2014). Malay Named Entity Recognition Based on Rule-Based Approach. *International Journal of Machine Learning and Computing*, *4*(3), 300–306. https://doi.org/10.7763/ijmlc.2014.v4.428

Attractions. - The world of Travel and Tourism. (n.d.). Retrieved May 18, 2021, from http://theworldoftandt.weebly.com/attractions.html

Borthwick, A. (1999). *A Maximum Entropy Approach to Named Entity Recognition*. Retrieved from http://www.cs.nyu.edu/web/Research/Theses/borthwick_andrew.pdf

Chantrapornchai, C., & Tunsakul, A. (2019). Information Extraction based on Named Entity for Tourism Corpus. *JCSSE 2019 - 16th International Joint Conference on Computer Science and Software Engineering: Knowledge Evolution Towards Singularity of Man-Machine Intelligence*, 187–192. https://doi.org/10.1109/JCSSE.2019.8864166

Chavan, A. (2019). Complete Tutorial on Named Entity Recognition (NER) using Python and Keras - AI, ML, Data Science Articles | Interviews | Insights | AI TIME JOURNAL. Retrieved April 9, 2021, from https://www.aitimejournal.com/@akshay.chavan/complete-tutorial-on-named-entity-recognition-ner-using-python-and-keras

Crystal, D. (1995). *The Cambridge encyclopedia of the English language*. Cambridge [England] ;;New York: Cambridge University Press.

Cui, Z., Ke, R., & Wang, Y. (2018). *Deep Bidirectional and Unidirectional LSTM Recurrent Neural Network for Network-wide Traffic Speed Prediction*. 1–11. Retrieved from http://arxiv.org/abs/1801.02143

Dogucu, M., & Çetinkaya-Rundel, M. (2020). *Web Scraping in the Statistics and Data Science Curriculum: Challenges and Opportunities Mine Dogucu & Mine Çetinkaya-Rundel Web Scraping in the Statistics and Data Science Curriculum: Challenges and Opportunities*. https://doi.org/10.1080/10691898.2020.1787116

Fan, H., Jiang, M., Xu, L., Zhu, H., Cheng, J., & Jiang, J. (2020). Comparison of Long Short Term Memory Networks and the Hydrological Model in Runoff Simulation. *Water*, *12*(1), 175. https://doi.org/10.3390/w12010175

Ge, Y., Xiong, H., Tuzhilin, A., & Liu, Q. (2014). Cost-Aware Collaborative Filtering for Travel Tour Recommendations. *ACM Trans. Inf. Syst*, *32*(4), 31. https://doi.org/10.1145/2559169

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), 184. https://doi.org/10.1186/s12859-016-1035-4

Huang, Z., Research, B., Xu, W., & Baidu, K. Y. (2015). *Bidirectional LSTM-CRF Models for Sequence Tagging*.

Ishino, A., Nanba, H., & Takezawa, T. (2011). Automatic Compilation of an Online Travel Portal from Automatically Extracted Travel Blog Entries. In *Information and Communication Technologies in Tourism 2011* (pp. 113–124). Springer Vienna. https://doi.org/10.1007/978-3-7091-0503-0_10

Kaur, D., & Gupta, V. (2010). A survey of Named Entity Recognition in English and other Indian Languages. *IJCSI International Journal of Computer Science Issues*, *7*(6). Retrieved from www.IJCSI.org

Lafferty, J., Mccallum, A., & Pereira, F. C. N. (2001). *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*. Retrieved from http://repository.upenn.edu/cis_papersPublisherURL:http://portal.acm.org/citation.cfm?id=655813PublisherURL:http://portal.acm.org/citation.cfm?id=655813Thisconferencepaperisavailableat ScholarlyCommons:http://repository.upenn.edu/cis_papers/159

Lyu, C., Chen, B., Ren, Y., & Ji, D. (2017). Long short-term memory RNN for biomedical named entity recognition. *BMC Bioinformatics*, *18*, 462. https://doi.org/10.1186/s12859-017-1868-5

Ma, X., & Hovy, E. (n.d.). *End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF*.

Manaswi, N. K. (2018). Deep Learning with Applications Using Python. In *Deep Learning with Applications Using Python*. Apress. https://doi.org/10.1007/978-1-4842-3516-4

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. Retrieved from http://ronan.collobert.com/senna/

Mikolov, T., & Le, Q. V. (n.d.). *Exploiting Similarities among Languages for Machine Translation*. Retrieved from https://code.google.com/p/word2vec/

Nayel, H. A., & Shashirekha, H. L. (2019). Integrating Dictionary Feature into A Deep Learning Model for Disease Named Entity Recognition. *ArXiv*, 1–16.

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

Sachan, D. S., Xie, P., Sachan, M., & Xing, E. P. (2018). Effective use of bidirectional language modeling for transfer learning in biomedical named entity recognition. *ArXiv*, 1–19.

Sarkar, D. (2018). Implementing Deep Learning Methods and Feature Engineering for Text Data: The Continuous Bag of Words (CBOW) - KDnuggets. Retrieved May 25, 2021, from https://www.kdnuggets.com/2018/04/implementing-deep-learning-methods-feature-engineering-text-data-cbow.html

Seok, M., Song, H.-J., Park, C.-Y., Kim, J.-D., & Kim, Y.-S. (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

Shen, Y., Yun, H., Lipton, Z. C., Kronrod, Y., & Anandkumar, A. (2018). Deep active learning for named entity recognition. *ArXiv*, 1–15. https://doi.org/10.18653/v1/W17-2630

Sobhana, N. ., Mitra, P., & Ghosh, S. K. (2010). Conditional Random Field Based Named Entity Recognition in Geological text. *International Journal of Computer Applications*, *1*(3), 143–147. https://doi.org/10.5120/72-166

Swarbrooke, J. (2002). *The Development and Management of Visitor Attractions*.

The 2014 Traveler's Road to Decision. (2014). Retrieved June 26, 2020, from https://www.thinkwithgoogle.com/consumer-insights/2014-travelers-road-to-decision/

Types of Attractions – Great Zim Traveller. (2016). Retrieved October 28, 2020, from https://www.greatzimtraveller.com/2016/02/types-of-attractions/

UNWTO. (2020). UNWTO World Tourism Barometer and Statistical Annex, January 2020. *UNWTO World Tourism Barometer*, *18*(1), 1–48. https://doi.org/10.18111/wtobarometereng.2020.18.1.1

Vijay, J., & Sridhar, R. (2016). A Machine Learning Approach to Named Entity Recognition for the Travel and Tourism Domain. *Asian Journal of Information Technology*. Retrieved from http://medwelljournals.com/abstract/?doi=ajit.2016.4309.4317

Visitor attractions - Travel and Tourism industry. (n.d.). Retrieved May 18, 2021, from http://stravelandtourism.weebly.com/visitor-attractions.html

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*(February), 103133. https://doi.org/10.1016/j.jbi.2019.103133

Wei, H., Gao, M., Zhou, A., Chen, F., Qu, W., Wang, C., & Lu, M. (2019). Named Entity Recognition from Biomedical Texts Using a Fusion Attention-Based BiLSTM-CRF. *IEEE Access*, *7*, 73627–73636. https://doi.org/10.1109/ACCESS.2019.2920734

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

Wilyawan, A. (2018). *Named Entity Recognition ( NER ) Bahasa Indonesia Menggunakan Conditional Random Field dan Pos-Tagging*.

Xue, L., Cao, H., Ye, F., & Qin, Y. (2019). A method of chinese tourism named entity recognition based on bblc model. *Proceedings - 2019 IEEE SmartWorld, Ubiquitous Intelligence and Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Internet of People and Smart City Innovation, SmartWorld/UIC/ATC/SCALCOM/IOP/SCI 2019*, 1722–1727. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00307

Zeng, D., Sun, C., Lin, L., & Liu, B. (2017). LSTM-CRF for Drug-Named Entity Recognition. *Entropy*, *19*(6). https://doi.org/10.3390/e19060283

Zhao, B. (2017). Web Scraping. In *Encyclopedia of Big Data* (pp. 1–3). Springer International Publishing. https://doi.org/10.1007/978-3-319-32001-4_483-1

# LAMPIRAN