LAPORAN PROYEK

Indonesian Named-entity Recognition Using BiLSTM-CRF

Disusun oleh:

1. 12S18014 – Giovanni Situmorang
2. 12S18015 – Della Cenovita Tarigan
3. 12S18022 – Alex Conro Manuel
4. 12S18042 – Indah Oktavia M. Sibarani
5. 12S18051 – Cindy Juniati Hutapea
6. 12S18062 – Ester Putri Dearest Sidabutar

11S4037 – PEMROSESAN BAHASA ALAMI

FAKULTAS INFORMATIKA DAN TEKNIK ELEKTRO

INSTITUT TEKNOLOGI DEL

2021

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

Pada bagian ini menyajikan latar belakang, tujuan, manfaat, dan ruang lingkup pengerjaan proyek.

1.1 Latar Belakang

Named Entity Recognition (NER) adalah salah satu topik penelitian di bidang *Natural Language Processing (NLP)*. Penelitian ini bertujuan untuk mengidentifikasi entitas bernama (*named entity*) kata di dalam kalimat [1]. Istilah "*Named Entity*" telah digunakan secara luas di bidang *information extraction (IE)*, *question answering (QA)*, dan berbagai bidang aplikasi *natural language processing (NLP)* lainnya. Istilah ini pertama kali digunakan di *Message Understanding Conference-6 (MUC-6)* pada tahun 1995 [3]. Terdapat 2 proses pengenalan entitas bernama dalam NER, yaitu identifikasi entitas kata yang tepat di dalam dokumen dan pengklasifikasian entitas-entitas tersebut ke dalam beberapa klasifikasi entitas umum dalam sistem NER seperti Organisasi, Orang, Lokasi, Tanggal, Waktu, Mata Uang, Persentase, Fasilitas dan *Geo Political Entities (GPE)* [2]. Tujuan dari NER adalah untuk mengidentifikasi atau mengklasifikasi sebuah entitas misalnya nama orang, organisasi, waktu, lokasi dan sesuatu entitas lain dalam sebuah teks yang sangat berguna dalam kasus ekstraksi informasi [7].

Penerapan NER dalam bahasa Inggris menunjukkan hasil yang baik dikarenakan banyaknya data yang tersedia dalam melakukan penelitian. Namun, penerapan NER dalam bahasa Indonesia tidak begitu baik, hal ini mungkin saja disebabkan oleh sifat bahasa Indonesia yang bervariasi dalam banyak hal karena sejarah morfologi [6]. Bahasa Indonesia memiliki karakteristik yang berbeda jika dibandingkan dengan bahasa Inggris. Bahasa Indonesia tidak memiliki indikasi gender seperti dalam bahasa Inggris. Bingung menentukan apakah kata ganti orang ketiga dalam bahasa Indonesia mengacu pada laki-laki atau perempuan. Kata ganti orang ketiga dalam bahasa Indonesia adalah-nya dan ia/dia. Ia/dia mewakili dia, dan-nya mewakili dia [5].

Pada penelitian yang dilakukan oleh Min Zhang, dkk, Penelitian ini, bertujuan untuk mengenali

beberapa jenis entitas peninggalan budaya, termasuk nama peninggalan budaya, dinasti peninggalan budaya), lokasi yang digali, dan koleksi museum, yang merupakan konsep kritis dalam penemuan pengetahuan peninggalan budaya. Namun, kurangnya data berlabel di bidang peninggalan budaya menyulitkan model pembelajaran mendalam yang mengandalkan data berlabel untuk mencapai kinerja yang sangat baik. Untuk mengatasi masalah ini, maka diusulkan model *deep learning semi supervised* bernama 'SCRNER (*Semi-supervised model for Cultural Relics' Named Entity Recognition*) yang memanfaatkan memori *bidirectional long short term* (BiLSTM) dan model *conditional random fields* (CRF) yang dilatih oleh data yang jarang diberi label dan banyak data tidak berlabel untuk mencapai kinerja yang efektif. *Bidirectional Long Short-Term Memory and Conditional Random Fields* (BiLSTM-CRF) pada penelitian ini dimulai dari data teks terlebih dahulu diubah ke dalam bentuk angka (*word embedding*). Word embedding dilakukan dengan menggunakan pendekatan ELMo. ELMo adalah kombinasi tugas khusus dari representasi lapisan menengah dalam bidirectional Language Model (biLM). Artinya, diberikan biLM yang telah dilatih sebelumnya dan arsitektur yang diawasi untuk tugas NLP target, model tugas akhir mempelajari kombinasi linier dari representasi lapisan.

Model bahasa yang dijelaskan di atas sepenuhnya agnostik tugas, dan dilatih dengan cara yang tidak diawasi. Hasil dari *word embedding* kemudian akan digunakan untuk melatih BiLSTM-CRF. Metode *Bidirectional Long Short-Term Memory* (BiLSTM) memanfaatkan informasi kontekstual masa lalu dan masa depan. Namun, dalam *sequence labeling* BiLSTM tidak menggeneralisasi korelasi antara label keluaran karena distribusi probabilitas BiLSTM tidak bergantung satu sama lain. *Sequence labeling* bertujuan untuk memberikan urutan dari label pada objek yang berurutan. Dalam *sequence labeling,* penting untuk mempertimbangkan korelasi antara label yang berdekatan. Oleh karena itu, pada penelitian ini model *sequence labeling* mengadopsi *Conditional Random Field* (CRF) di lapisan atas untuk memprediksi label dari semua kalimat secara bersamaan. *Conditional Random Field* (CRF) memperhitungkan letak label yang muncul sebelumnya kemudian setiap fitur (label) akan diberikan bobot dengan tujuan menghitung setiap probabilitas untuk label berikutnya [4].

Secara umum, terdapat empat tantangan atau permasalahan NER dalam proses pengenalan entitas bernama, yaitu [1]:

1. Deteksi batasan dari entitas bernama (NE *boundary detection*), jika entitas bernama berupa gabungan dua atau lebih kata.
2. Homonim, kata yang memiliki lafal dan ejaan yang sama dengan kata lain tetapi berbeda maknanya. Kedua kata ini memungkinkan untuk memiliki entitas bernama yang berbeda.
3. *Spelling mistakes*, biasanya permasalahan ini ada jika teks merupakan hasil dari forum diskusi online, bahasa yang tidak formal dan kurang terstruktur.
4. Struktur kalimat yang tidak baku: Struktur kalimat adalah penting untuk penandaan *part-of-speech*, yang digunakan untuk mendeteksi *frase* kata benda dari teks, untuk membantu dalam deteksi batas

Selain permasalah diatas, masalah yang sering muncul pada *named-entity recognition* adalah *Ambiguity*. Pertama, kata yang sama dapat berarti dua entitas yang berbeda. Misalnya kata Soekarno dapat berarti presiden pertama Indonesia, atau nama belakang seorang seniman (Enrico Soekarno), keduanya entitas berbeda walaupun tipenya sama (orang/*person*). Jenis ambiguitas kedua adalah nama yang sama tapi tipe berbeda. Contohnya adalah Bung Karno sebagai stadion dengan Bung Karno sebagai orang. Berdasarkan permasalahan tersebut maka dibutuhkan sebuah metode yang dapat mengatasi permasalahan diatas.

Pada penelitian ini, metode yang digunakan adalah metode *Bidirectional* LSTM-CRF. *Bidirectional Long Short-Term Memory* (BiLSTM) mempelajari bobot keluaran dari konteks sebelumnya dan masukan dari setiap urutan pada saat berlangsung [4]. *Conditional Random Field* merupakan sebuah model probabilistik yang digunakan untuk memprediksi data terstruktur yang telah digunakan dalam berbagai tugas, seperti *natural language processing* [7]. Model CRF melakukan training untuk memprediksi sebuah vektor dari sebuah kalimat. *Conditional random fields* (CRFs) telah terbukti efektif di banyak area pemrosesan bahasa alami (NLP), termasuk tugas tag urutan dan pengenalan entitas bernama (NER) [4]. ELMo (*Embeddings from Language Models*) merupakan *contextualized word embeddings*, yaitu model yang memperhitungkan konteks kata ketika mengubah kalimat ke dalam bentuk vektor, sehingga hasilnya lebih akurat dan relevan. ELMo menerima *input* berupa kalimat dan *output* berupa hasil vektor perhitungan setiap kata dalam bentuk 1024 dimensi, penggunaan pre-trained model ELMo terbukti dapat meningkatkan akurasi dalam berbagai kegiatan NLP termasuk *Named Entity Recognation*. Oleh

karena itu, dengan mengkombinasikan pendekatan *bidirectional* LSTM dan model CRF serta ELMo diharapkan dapat digunakan untuk menangkap semua informasi selama pemodelan urutan kalimat panjang. Berdasarkan uraian tersebut maka pada penelitian ini akan dilakukan Implementasi pendekatan *Bidirectional* LSTM-CRF dengan model ELMO pada kasus *Named Entity Recognition* dalam teks bahasa Indonesia dengan batasan data yang akan digunakan pada penelitian ini adalah *dataset* singgalang.

1.2 Tujuan

Adapun tujuan dari pengerjaan proyek ini adalah

1. Membuat model *Indonesian Named Entity Recognition* yang dapat melakukan pengenalan pada teks bahasa indonesia serta mendeteksi dan mengklasifikasikan kata kedalam entitas yang seharusnya.
2. Untuk mengetahui tingkat akurasi penggunaan metode BiLSTM-CRF dalam membangun model NER untuk bahasa Indonesia.

1.3 Manfaat

Manfaat dari pengerjaan proyek ini yaitu:

1. Proyek ini diharapkan nantinya dapat memberi wawasan serta pengetahuan bagi tim proyek dalam menerapkan aplikasi pemrosesan bahasa alami khususnya *named entity recognition* pada bahasa Indonesia
2. Proyek ini diharapkan dapat digunakan dalam membantu mendeteksi informasi penting dan mengidentifikasi elemen kunci dalam teks dengan mudah dan tepat sesuai entitasnya, seperti nama orang, tempat, dan lainnya.

1.4 Ruang Lingkup

Ruang lingkup yang akan dibahas dalam proyek ini adalah model yang dibangun hanya bisa menerima *input* berupa kalimat Bahasa Indonesia. Metode yang digunakan ialah bidirectional long short term (BiLSTM) dan model conditional random fields (CRF) serta Feature selection dengan menggunakan ELMo. Dataset yang digunakan adalah SINGGALANG.tsv yang berasal dari github yaitu: https://github.com/ialfina/ner-dataset-modified-dee/tree/master/singgalang.

BAB 2
ISI

Bab ini menjelaskan mengenai analisis yang akan dilakukan terhadap data dan metode, desain pemrosesan bahasa alami dan implementasi berupa kode program dan cuplikan hasil, dan hasil evaluasi kuantitatif terhadap implementasi pemrosesan bahasa alami yang dilakukan.

2.1 Analisis

Pada sub bab ini dijelaskan mengenai metode yang akan digunakan dan analisis yang dilakukan terhadap data. Analisis yang dilakukan terdiri dari analisis sumber data, analisis bentuk data, analisis metode *text preprocessing*, analisis metode *bidirectional* LSTM-CRF, dan analisis metode evaluasi penelitian.

2.1.1 Analisis Data

Dataset yang digunakan dalam proyek ini dapat dilihat pada link berikut: https://github.com/ialfina/ner-dataset-modified-dee/tree/master/singgalang. *Dataset* yang digunakan adalah SINGGALANG.tsv, *dataset* tersebut terdiri dari 1478268 *records*, dimana *data train* terdiri dari 80% dari *dataset* dan data *test* terdiri dari 20 % dari *dataset*. *Dataset* tersebut telah diberi label yaitu keterangan entitas pada setiap kata dalam *dataset* tersebut. Atribut yang terdapat dalam dataset ialah 2 atribut berupa 0 dan 1.

Tabel 1 Atribut pada Dataset

No	Nama Atribut	Tipe Atribut	Deskripsi
1.	0	Nominal	Kamus kata dalam bahasa Indonesia
2.	1	Nominal	Keterangan entitas kata

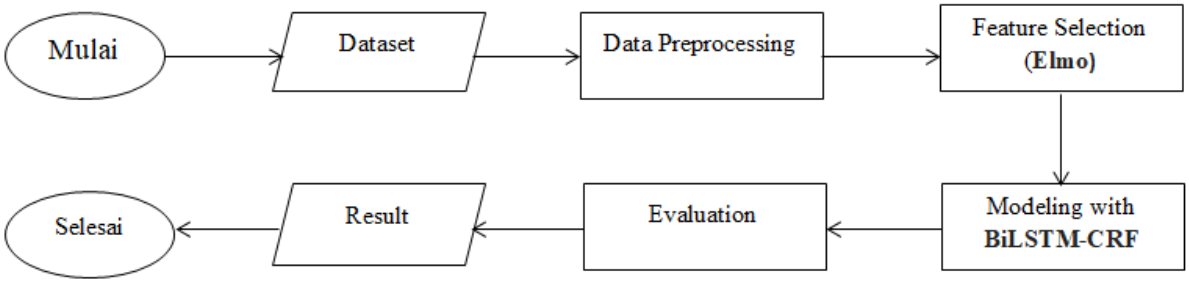
2.1.2 Analisis Metode

Named Entity Recognition (NER) atau Pengenalan entitas bernama merupakan salah satu pemrosesan data yang banyak digunakan dan juga merupakan bagian dari NLP. *Named Entity Recognition* akan mengidentifikasi kunci-kunci informasi dalam sebuah teks dan

mengklasifikasikannya ke dalam satu set kategori yang telah ditentukan. Pada dasarnya entitas merupakan hal yang secara konsisten dibicarakan dalam teks. Langkah-langkah yang dilakukan dalam proses NER ialah mendeteksi entitas dari teks dan mengklasifikasikan entitas tersebut kedalam kategori tertentu seperti, *Person*, *Organization* atau *Place/Location*. Dalam melakukan klasifikasi kategori kata kunci dalam sebuah teks diperlukan suatu metode atau algoritma klasifikasi.

2.2 Desain

Pada subbab ini akan menjelaskan desain pemrosesan bahasa alami yaitu *named entity recognition* pada bahasa Indonesia yang ditampilkan dalam bentuk *flowchart* atau diagram alir seperti ditunjukkan pada gambar berikut.



Gambar 1 Desain Analisis *Indonesian Named Entity Recognition*

2.2.1 Data Preprocessing

Data Preprocessing merupakan tahapan awal pada pemrosesan yang dilakukan sebelum teks (*dataset*) diolah ke tahap selanjutnya. Data yang diperoleh masih dalam format yang tidak terstruktur, dimana masih terdapat *noise* pada *dataset* tersebut. Oleh karena itu perlu dilakukan *data preprocessing* untuk menghilangkan kata-kata pada teks atau dokumen yang mengandung beberapa format yang keberadaannya tidak penting dalam *text mining*.

2.2.1.1 Data Cleaning

Data cleaning adalah salah satu tahapan dalam data preparation. Hampir semua *dataset* yang akan digunakan untuk *modelling* tidak dalam kondisi siap pakai, misalnya string yang kosong *(incomplete data/missing value)* untuk itu perlu dilakukan pengecekan data dan membersihkan

dan memperbaiki struktur data yang dimiliki. Adapun beberapa cara yang dapat dilakukan untuk membersihkan data adalah sebagai berikut:

a. Mengabaikan data yang hilang dengan cara menghapus data tersebut, namun cara ini tidak cukup efektif ketika terdapat banyak data yang hilang.
b. Mengidentifikasi dan menghapus data yang duplikat, data yang memiliki nilai yang sama persis untuk tiap kolomnya kurang berguna untuk dipertahankan. Data duplikat juga sangat mungkin akan mempengaruhi akurasi model
c. Mengisi *missing value*, namun cara ini tidak cukup efektif ketika datanya besar dan membutuhkan waktu yang lama.
d. Memperbaiki penulisan kata yang kurang tepat.
e. Menghapus karakter-karakter yang bersifat noise pada *dataset*.
f. Hapus kolom yang hanya memiliki satu nilai
g. Menghapus kolom yang hanya memiliki satu nilai atau nilai tunggal, Kolom yang hanya memiliki satu nilai sejatinya tidak akan memberikan dampak yang signifikan dan mungkin tidak berguna untuk proses *modelling*.

2.2.1.2 Expand Contraction

Tujuan dari tahapan ini adalah untuk memperbaiki format penulisan pada kolom dalam dataset dengan menghilangkan apostrof dan menuliskan kata-kata secara langsung tanpa menggunakan singkatan.

2.2.1.3 Case Folding

Case Folding adalah teknik pemrosesan teks yang mengubah semua huruf menjadi *case* yang sama. Dalam penelitian ini semua huruf akan diubah menjadi *lowercase* atau huruf kecil. Hal ini dilakukan agar mudah bagi mesin untuk menafsirkan kata-kata karena huruf kecil dan huruf besar diperlakukan berbeda oleh mesin. Tujuan dari *Case Folding* ini adalah agar kata-kata yang sama tidak terdeteksi berbeda hanya karena perbedaan terdapat huruf kapital.

Tabel 2 Contoh Case Folding

Text	***Case Folding***
Presiden	presiden

2.2.1.4 Anotasi Bio

Anotasi BIO merupakan sebuah proses untuk memberikan tag terhadap setiap kata dalam dataset dengan menambahkan kolom baru untuk mengklasifikasikan setiap kata ke dalam tag tertentu.

2.2.2 Feature Selection with ELMo

Feature Selection merupakan salah satu kegiatan yang umumnya bisa dilakukan secara preprocessing dan bertujuan untuk memilih *feature* yang berpengaruh dan mengesampingkan *feature* yang tidak berpengaruh dalam suatu kegiatan pemodelan atau penganalisaan data. Dalam pengerjaan proyek ini, akan dilakukan *feature selection* dengan menggunakan Elmo.

2.2.3 Modeling with BiLSTM-CRF

Setelah tahapan *feature selection* selesai dilakukan, pada *dataset* akan dilakukan train dengan menggunakan Metode *Bidirectional Long Short-Term Memory* (BiLSTM) dan model *Conditional Random Fields* (CRF). *Pada* penelitian ini dimulai dari data teks terlebih dahulu diubah ke dalam bentuk angka (word embedding). *Bidirectional* LSTM menggabungkan konteks sebelumnya dan konteks setelahnya dengan memproses data dari dua arah yang selanjutnya diklasifikasi menggunakan *Conditional Random Fields* (CRF).

Bidirectional Long Short-Term Memory (BiLSTM) mempelajari bobot keluaran dari konteks sebelumnya dan masukan dari setiap urutan pada saat berlangsung. Informasi dari kata sebelumnya dan kata selanjutnya dari urutan kalimat dapat ditangkap secara bersamaan oleh *forward network* yang merepresentasikan konteks sebelumnya dan *backward network* yang merepresentasikan konteks setelahnya yang diikuti dengan memproses data dari dua arah dengan *hidden layer*, sehingga memperoleh informasi konteks dalam proses pemodelan urutan kalimat. *Conditional Random Field* merupakan sebuah model probabilistik yang digunakan untuk memprediksi data terstruktur yang telah digunakan dalam berbagai tugas, seperti *natural language processing*. Model CRF melakukan *training* untuk memprediksi sebuah vektor dari sebuah kalimat.

Dalam *sequence labeling*, BiLSTM tidak menggeneralisasi korelasi antara label keluaran karena distribusi probabilitas BiLSTM tidak bergantung satu sama lain. *Sequence labeling* bertujuan untuk memberikan urutan dari label pada objek yang berurutan. Dalam sequence labeling, penting untuk mempertimbangkan korelasi antara label yang berdekatan. Oleh karena itu, pada penelitian ini model *sequence labeling* mengadopsi *Conditional Random Fields* (CRF) di lapisan atas untuk memprediksi label dari semua kalimat secara bersamaan. *Conditional Random Fields* (CRF) memperhitungkan letak label yang muncul sebelumnya kemudian setiap fitur (label) akan diberikan bobot dengan tujuan menghitung setiap probabilitas untuk label berikutnya.

2.2.4 Evaluation and Result

Untuk menentukan model terbaik, perlu suatu teknik yang digunakan untuk mengukur kinerja suatu model khususnya kasus klasifikasi adalah *confusion matrix. Confusion matrix* adalah salah satu tools analitik prediktif yang menampilkan dan membandingkan nilai aktual atau nilai sebenarnya dengan nilai hasil prediksi model yang dapat digunakan untuk menghasilkan matrik evaluasi seperti *Accuracy* (akurasi), *Precision, Recall, dan F1-Score atau F-Measure*. Ada empat nilai yang dihasilkan di dalam tabel *confusion matrix*, di antaranya *True Positive (TP), False Positive (FP), False Negative (FN),* dan *True Negative (TN)*. Ilustrasi tabel *confusion matrix* dapat dilihat pada gambar berikut.

Tabel 3 *Confusion Matrix*

	Positive	**Negative**
Positive	TP	FP
Negative	FN	TP

Keterangan:

True Positive (TP): Jumlah data yang bernilai Positif dan diprediksi benar sebagai Positif.

False Positive (FP): Jumlah data yang bernilai Negatif tetapi diprediksi sebagai Positif.

False Negative (FN): Jumlah data yang bernilai Positif tetapi diprediksi sebagai Negatif.

True Negative (TN): Jumlah data yang bernilai Negatif dan diprediksi benar sebagai Negatif.

Cara menghitung metode evaluasi tersebut dengan menggunakan *confusion matrix*:

1) Accuracy

Nilai akurasi didapatkan dari jumlah data bernilai positif yang diprediksi positif dan data bernilai negatif yang diprediksi negatif dibagi dengan jumlah seluruh data di dalam dataset.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

2) Precision

Precision adalah peluang kasus yang diprediksi positif yang pada kenyataannya termasuk kasus kategori positif.

$$Precision = \frac{TP}{TP + FP}$$

3) Recall

Recall adalah peluang kasus dengan kategori positif yang dengan tepat diprediksi positif.

$$Recall = \frac{TP}{TP + FN}$$

4) F1-Score atau *F-Measure*

Nilai *F1-Score* atau dikenal juga dengan nama *F-Measure* didapatkan dari hasil *Precision* dan *Recall* antara kategori hasil prediksi dengan kategori sebenarnya.

$$F1 = \frac{2\ x\ Recall\ x\ Precision}{Recall + Precision}$$

2.3 Implementasi

Pada subbab ini dijelaskan mengimplementasikan pemrosesan bahasa alami, yaitu *sentiment analysis* menggunakan algoritma *Bidirectional Long Short-Term Memory* (Bi-LSTM).

2.3.1 Data Preprocessing

Pada bagian ini akan dibahas *data preprocessing* yang dilakukan sebelum digunakan dalam pemodelan, mencakup *data cleaning, expand contraction, case folding,* dan Anotasi Bio.

2.3.1.1 Data cleaning

Data cleaning merupakan salah satu tahapan dalam *data preparation* dimana pada fase ini dilakukan pembersihan data, misalnya string yang kosong (*incomplete data*/*missing value*).

Pada bagian ini dilakukan pengecekan terhadap *dataset* untuk mengetahui apakah *dataset* yang digunakan memiliki *missing value* atau tidak memiliki nilai sehingga ketika ditemukan *missing value* maka ditentukan solusi untuk mengatasinya. Pengecekan yang dilakukan dengan menggunakan beberapa fungsi, diantaranya Fungsi `isna()` berguna untuk mengembalikan nilai boolean (True dan False). Dimana apabila *cell* berisi *value* "False", maka artinya *cell* tersebut tidak mengandung *missing value* dan sebaliknya, jika *cell* berisi *value* "True", maka *cell* tersebut mengandung *missing value*. Berikut adalah kode program yang digunakan untuk mengecek missing value.

```
#Cek Missing Value
df.isna()
```

Gambar 2 Kode Program Mengecek *Missing Value*

Hasil yang diperoleh dalam pengecekan *missing value* ditunjukkan pada gambar berikut.

	0	1
0	False	False
1	False	False
2	False	False
3	False	False
4	False	False
...	...	...
1478263	False	False
1478264	False	False
1478265	False	False
1478266	False	False
1478267	False	False

1478268 rows × 2 columns

Gambar 3 Hasil Pengecekan Missing Value

Untuk memudahkan dalam memahami data, maka perlu dilakukan agregasi data dengan fungsi `sum()`. Dengan menggunakan fungsi `sum()`, maka akan diketahui berapa jumlah data yang *missing value* dan berasal dari atribut apa. Pengecekan jumlah nilai yang null pada setiap variabel dengan perintah `df.isna().sum()` dan ditemukan terdapat 20 nilai yang null pada atribut 0. Berikut adalah kode program yang digunakan untuk menghitung jumlah missing value.

```
#Hitung Jumlah Missing Value
df.isna().sum()
```

Gambar 4 Kode Program Menghitung Jumlah Missing Value

Hasil yang diperoleh dalam menghitung jumlah *missing value* ditunjukkan pada gambar berikut.

```
0    20
1     0
dtype: int64
```

Gambar 5 Hasil Menghitung Jumlah Missing Value

Untuk mengatasi masalah tersebut maka perlu dilakukan penghapusan terhadap data yang bernilai null. Hal ini didasarkan pada jumlah data yang null hanya sedikit dibandingkan dengan *record data* pada *dataset* dan tidak memungkinkan juga untuk dilakukan pengisian nilai kosong dengan rata-rata. Selanjutnya menggunakan fungsi `.dropna()` untuk menghapus data yang

bernilai *null*. Setelah fungsi `.dropna()` dijalankan, maka data yang mengandung *missing value* terhapus. Pada gambar dibawah menunjukkan bahwa pada *dataset* sudah tidak terdapat *missing value*.

```
#Drop Missing Value
df.dropna(inplace=True)
df.isnull().sum()
```

Gambar 6 Kode Program Menghapus *Missing value*

Hasil yang diperoleh setelah menghapus *missing value* ditunjukkan pada gambar berikut.

```
0    0
1    0
dtype: int64
```

Gambar 7 Hasil Menghapus *Missing value*

2.3.1.2 *Expand Contraction*

Pada tahap ini akan dilakukan proses *expand contraction* untuk memperbaiki format penulisan pada kolom dalam dataset. Berikut adalah kode program yang digunakan untuk melakukan expand contraction pada data train data dan test data.

```
# Memperbaiki format penulisan pada kolom token
def perbaikan_format_tanda_baca(dataset_prep):
    for i in dataset_prep.index:
        token = re.sub('([.,!?()])', r'\1 ', dataset_prep.at[i,
'token'])
        dataset_prep.at[i, 'token'] = re.sub('\s{2,}', ' ', token)
    return dataset_prep
perbaikan_format_tanda_baca(dataset_prep)
```

Gambar 8 Kode Program Expand Contraction

Hasil perbaikan format penulisan kolom token ditunjukkan pada gambar berikut.

	token	entitas bernama	kalimat
0	Ia	O	1
1	menjabat	O	1
2	sebagai	O	1
3	Presiden	O	1
4	ketiga	O	1
...	...	...	...
1478263	di	O	48957
1478264	kalangan	O	48957
1478265	pelajar	O	48957
1478266	Muhammadiyah	Organisation	48957
1478267	.	O	48957

1478268 rows × 3 columns

Gambar 9 Hasil *Expand Contraction*

Selanjutnya mendefinisikan dictionary yang digunakan yaitu berupa singkatan maupun akronim, yang kemudian menampilkan singkatan tersebut pada dataset. Berikut adalah kode program yang digunakan untuk melakukan menampilkan singkatan pada dataset.

```
#menampilkan singkatan pada dataset
dataset_prep[dataset_prep.token.str.contains('|'.join(contractions_dic
t.keys()), regex=True)]
```

Gambar 10 Kode Program Menampilkan Singkatan Pada Dataset

Hasil singkatan maupun akronim yang terdapat dalam dataset ditunjukkan pada gambar berikut.

	token	entitas bernama	kalimat
412	W. H.	O	14
2716	Dr.	O	94
2862	Mr.	O	100
4737	H.	O	158
4885	H.	O	163
...	...	...	...
1472406	St.	O	48758
1474016	S. H.	O	48810
1474035	S. H.	O	48810
1474331	Dr.	O	48822
1475058	H.	O	48848

1760 rows × 3 columns

Gambar 11 Hasil Menampilkan Singkatan Pada *Dataset*

Selanjutnya melakukan expand contraction, dengan membuat data frame yang berisi token sebelum dan sesudah di expand yang selanjutnya menampilkan data tertentu. Berikut adalah kode program yang digunakan untuk expand contraction.

```
#untuk expand contraction
#inplace true = the data is modified in place
dataset_prep.token.replace(to_replace=contractions_dict, inplace=True,
value=None, method=None, regex=True)
```

Gambar 12 Kode Program Expand Contraction

```
#Membuat data frame yang berisi token sebelum dan setelah di expand
# untuk menggabungkan kedua dataframe
df_result = pd.concat([dataset, dataset_prep], axis=1, join='inner')

# untuk menghapus kolom kalimat dan entitas bernama
df_result = df_result.drop(columns=['entitas bernama', 'kalimat'])

# untuk rename nama kolom
```

```
df_result.columns.values[0] = "before_expand"
df_result.columns.values[1] = "after_expand"
# menampilkan token sebelum dan setelah di expand
display(df_result.loc[[4740,392707,1329234], :])
```

Gambar 13 Kode Program Memperbaiki Dataset dengan Expand Contraction

Hasil expand contraction yang terdapat dalam dataset ditunjukkan pada gambar berikut.

	before_expand	after_expand
4740	ER	ER
392707	,	,
1329234	diterbitkan	diterbitkan

Gambar 14 Hasil Expand Contraction

2.3.1.3 Case Folding

Pada bagian ini dilakukan untuk mengubah isi dataset yang sudah dibersihkan ke dalam format huruf kecil (lowercase). Berikut adalah kode program yang dilakukan untuk mengubah isi dataset yang sudah dibersihkan menjadi dalam format lowercase.

```
# lower()=mengubah upper case to lower case
dataset_prep["token"] = dataset_prep["token"].str.lower()
dataset_prep.head()
```

Gambar 15 Kode Program Case Folding

Hasil case folding dari dataset yang sudah dibersihkan sebelumnya ditunjukkan pada Gambar 16.

	token	entitas bernama	kalimat
0	ia	O	1
1	menjabat	O	1
2	sebagai	O	1
3	presiden	O	1
4	ketiga	O	1

Gambar 16 Hasil Case Folding

2.3.1.3 Anotasi Bio

Pada bagian ini dilakukan anotasi bio yang merupakan proses untuk memberikan tag terhadap setiap kata dalam dataset dengan menambahkan kolom baru, yaitu bio_tag untuk mengklasifikasikan setiap kata ke dalam tag tertentu. Berikut adalah kode program yang dilakukan untuk menghasilkan anotasi BIO.

```
bio_tag = []
prev_tag = "O"
for tag in dataset_prep['entitas bernama']:
    if tag == "O": #O
      bio_tag.append((tag))
      prev_tag = tag
      continue
    if tag != "O" and prev_tag == "O": # Begin NE
      bio_tag.append(("B-"+tag))
      prev_tag = tag
    elif prev_tag != "O" and prev_tag == tag: # Inside NE
      bio_tag.append(("I-"+tag))
      prev_tag = tag
    elif prev_tag != "O" and prev_tag != tag: # NE yang berdekatan
      bio_tag.append(( "B-"+tag))
      prev_tag = tag
```

Gambar 17 Kode Program Mendefenisikan Anotasi Bio

```
#Menggabungkan kolom bio_tag ke dataset
dataset_prep['bio_tag'] = bio_tag
dataset_prep.iloc[:10279]
```

Gambar 18 Kode Program Menggabungkan kolom bio_tag ke dataset

Hasil anotasi BIO berupa gabungan kolom bio_tag ke dalam dataset, ditunjukkan pada Gambar 19.

	token	entitas bernama	kalimat	bio_tag
0	ia	O	1	O
1	menjabat	O	1	O
2	sebagai	O	1	O
3	presiden	O	1	O
4	ketiga	O	1	O
...	...	...	...	...
10274	,	O	338	O
10275	binduriang	Place	338	B-Place
10276	,	O	338	O
10277	sindang	Place	338	B-Place
10278	dataran	Place	338	I-Place

10279 rows × 4 columns

Gambar 19 Hasil Anotasi BIO

```
#menghapus kolom entitas bernama
del dataset_prep['entitas bernama']
```

Gambar 20 Menghapus Kolom Entitas Bernama

```
#rename name kolom bio_tag menjadi entitas bernama
dataset_prep.rename(columns={'bio_tag':'entitas bernama'},
inplace=True)
```

Gambar 21 Kode Program Rename Kolom bio_tag menjadi Entitas Bernama

```
# Merubah posisi kolom
dataset_prep = dataset_prep[['token', 'entitas bernama', 'kalimat']]
cols = list(dataset_prep.columns.values)
```

Gambar 22 Kode Program Merubah Posisi Kolom

```
#Memanggil dataset hasil preprocessing
ner_dataset = pd.read_csv('/content/ner_dataset.tsv')
ner_dataset.head())
```

Gambar 23 Kode Program Memanggil Dataset Hasil Preprocessing

Berikut adalah hasil dari dataset yang sudah dilakukan data preprocessing.

	token	entitas bernama	no_kalimat
0	ia	O	1
1	menjabat	O	1
2	sebagai	O	1
3	presiden	O	1
4	ketiga	O	1

Gambar 24 Hasil Data Preprocessing

2.3.2 Feature Selection ELMo

Pada tahap ini akan dilakukan proses *feature selection* menggunakan ELMo (*Embeddings from Language Models*). ELMo bertujuan untuk memberikan representasi kata yang lebih baik untuk tugas NLP dalam konteks yang berbeda dengan menghasilkan beberapa *embeddings* kata per kata, di berbagai skenario. ELMo menggunakan model bahasa dua arah (biLM) untuk mempelajari konteks kata dan linguistik. Pada setiap kata, status internal dari *forward* dan *backward* pass digabungkan untuk menghasilkan vektor kata perantara. Dengan demikian, sifat dua arah model yang memberikan petunjuk tidak hanya untuk kata berikutnya dalam kalimat tetapi juga kata-kata yang muncul sebelumnya. Representasi akhir (ELMo) adalah jumlah bibit dari vektor kata mentah dan 2 vektor kata perantara. Berikut adalah kode program yang dilakukan untuk melakukan *feature selection* menggunakan ELMo.

```
import tensorflow as tf
import tensorflow_hub as hub
from tensorflow.keras import backend as K
sess = tf.Session()
K.set_session(sess)
```

Gambar 25 Kode Program Import Library ELMo

Pada proyek, kami menggunakan *hyperparameter* dengan nilai 32 sehingga jumlah sampel harus dapat dibagi sesuai dengan nilai *hyperparameter* agar tidak terjadi *error*. Berikut code programnya.

```
batch_size = 32
X_tr, X_val = X_tr[:1110*batch_size], X_tr[-135*batch_size:]
y_tr, y_val = y_tr[:1110*batch_size], y_tr[-135*batch_size:]
y_tr = y_tr.reshape(y_tr.shape[0], y_tr.shape[1], 1)
y_val = y_val.reshape(y_val.shape[0], y_val.shape[1], 1)
```

Gambar 26 Kode Program Hyperparameter dengan Nilai 32

```
# Untuk mendapatkan representasi kata pada semua kalimat
#https://stackoverflow.com/questions/55902437/elmo-embedding-start-
session
# https://tfhub.dev/google/elmo/3
# elmo akan menghasilkan vektor dengan panjang 1024 dengan jumlah
kalimat adalah 44.000
def ElmoEmbedding(x):
    return elmo_model(inputs={
                            "tokens": tf.squeeze(tf.cast(x,
tf.string)),
                            "sequence_len":
tf.constant(batch_size*[max_len])
                      },
                      signature="tokens",
                      as_dict=True)["elmo"]
```

Gambar 27 Kode Program Mendefenisikan ElmoEmbedding

```
# Karena tidak memungkinkan untuk menampilkan semua representasi kata
#  Untuk melihat contoh  vektor pada kalimat index ke 0
def elmo_vectors(x):
    embeddings = elmo_model(x, signature="default",
as_dict=True)["elmo"]
    with tf.Session() as sess:
        sess.run(tf.global_variables_initializer())
        sess.run(tf.tables_initializer())
        return sess.run(embeddings)
v = elmo_vectors(X[0])
print(v.shape)
print(v[0][0]) # untuk menampilkan vektor
```

Gambar 28 Kode Program Mendefenisikan elmo_vectors

```
print(X[0])
```

Gambar 29 Kode Program Menampilkan Kalimat Pertama

```
['ia', 'menjabat', 'sebagai', 'presiden', 'ketiga', 'mesir', 'pada', 'periode', '15', 'oktober', '1970', 'hingga', 'terbunuhnya', '
```

Gambar 30 Hasil Model ELMo Untuk Menampilkan Kalimat Pertama

2.3.3 Modeling with BiLSTM

Pada bagian ini dilakukan pemodelan terhadap penamaan entitas dalam bahasa Indonesia dengan algoritma Bidirectional LSTM. Berikut adalah kode program yang digunakan, ditunjukkan pada gambar 31.

```
input_text = Input(shape=(max_len,), dtype=tf.string)
embedding = Lambda(ElmoEmbedding, output_shape=(max_len,
1024))(input_text)
x = Bidirectional(LSTM(units=512, return_sequences=True,
                       recurrent_dropout=0.2, dropout=0.2))(embedding)
x_rnn = Bidirectional(LSTM(units=512, return_sequences=True,
                           recurrent_dropout=0.2, dropout=0.2))(x)
x = add([x, x_rnn])  # residual connection to the first biLSTM
out = TimeDistributed(Dense(n_tags, activation="softmax"))(x)
tag_model = Model(input_text, out)
tag_model.compile(optimizer="adam",
loss="sparse_categorical_crossentropy", metrics=["accuracy"])
```

Gambar 31 Kode Program Transfer Model Agar Fit

```
# fit the model
# fit the model, or not?

if SAVE_OR_NOT == True:
    csv_logger = CSVLogger(path + 'ep_1.csv',
                           separator=',',
                           append=False)
    tag_model.fit(np.array(X_tr), y_tr,
              validation_data=(np.array(X_val), y_val),
```

```
              epochs=1,
              batch_size=32,
              verbose=1,
              callbacks=[csv_logger])
elif SAVE_OR_NOT == False:
    print('skipped')
```

Gambar 32 Kode Program Pengecekan Model (Fit)

2.4 Hasil

Pada subbab ini dijelaskan mengenai evaluasi berdasarkan implementasi pemrosesan bahasa alami, yaitu Indonesian Named-entity Recognition menggunakan BiLSTM-CRF Untuk mengukur performance metrics dilakukan dengan menghitung nilai precision, Recall, dan F1-Score.

1. Precision menggambarkan tingkat keakuratan antara data yang diminta dengan hasil prediksi yang diberikan oleh model.

$$Precision = \frac{TP}{TP + FP}$$

2. Recall menggambarkan keberhasilan model dalam menemukan kembali sebuah informasi.

$$Recall = \frac{TP}{TP + FN}$$

3. F1-Score menggambarkan harmonic *mean* dari *precision* dan *recall*.

$$\frac{1}{F1} = \frac{1}{2}\left(\frac{1}{precision} + \frac{1}{recall}\right)$$

2.4.1 Evaluation BiLSTM Model - Elmo

Berikut adalah kode program evaluasi terhadap model BiLSTM yang telah dibangun dengan Elmo.

```
from seqeval.metrics import accuracy_score, precision_score,
recall_score, f1_score, classification_report
X_te = X_te[:149*batch_size]
test_pred = tag_model.predict(np.array(X_te), verbose=1)
```

Gambar 33 Kode Program Mendefenisikan Confusion Matrix

```
idx2tag = {i: w for w, i in tags2index.items()}

def pred2label(pred):
    out = []
    for pred_i in pred:
        out_i = []
        for p in pred_i:
            p_i = np.argmax(p)
            out_i.append(idx2tag[p_i].replace("PAD", "O"))
        out.append(out_i)
    return out

def test2label(pred):
    out = []
    for pred_i in pred:
        out_i = []
        for p in pred_i:
            out_i.append(idx2tag[p].replace("PAD", "O"))
        out.append(out_i)
    return out

pred_labels = pred2label(test_pred)
test_labels = test2label(y_te[:149*32])
```

Gambar 34 Kode Program Mendefenisikan data train dan data test

Berikut adalah kode program evaluasi berupa nilai akurasi untuk model yang telah dibangun.

```
print("accuracy: {:.1%}".format(accuracy_score(test_labels,
pred_labels)))
#menyajikan dalam bentuk koma
# print("F1-score: ", f1_score(test_labels, pred_labels))
```

Gambar 35 Kode Program Mendapatkan Nilai Akurasi

Hasil nilai akurasi BiLSTM dengan ELMo yang didapatkan didapatkan pada gambar berikut. Hasil menunjukkan bahwa nilai akurasinya adalah 95,9%

```
accuracy: 95.9%
```

Gambar 36 Hasil Nilai Akurasi

Berikut adalah kode program evaluasi berupa confusion matrix untuk model yang telah dibangun.

```
print(classification_report(test_labels, pred_labels))
```

Gambar 37 Kode Program Mendapatkan Nilai Confusion Matrix

Hasil nilai confusion matrix yang didapatkan ditunjukkan pada gambar berikut.

	precision	recall	f1-score	support
Organisation	0.37	0.14	0.21	685
Person	0.34	0.11	0.17	2703
Place	0.72	0.40	0.51	5654
micro avg	0.62	0.29	0.40	9042
macro avg	0.48	0.22	0.30	9042
weighted avg	0.58	0.29	0.39	9042

Gambar 38 Hasil Nilai Confusion Matrix

BAB 3
PENUTUP

Pada bab ini dijelaskan mengenai pembagian tugas dan tanggung jawab dalam pengerjaan proyek, kesimpulan yang diperoleh, dan saran terhadap proyek ke depannya.

3.1 Pembagian Tugas dan Tanggung Jawab

Pada subbab ini dijelaskan pembagian tugas dan tanggung jawab dari setiap anggota dalam pengerjaan proyek.

Tabel 4 Pembagian Tugas dan Tanggungjawab

NIM/Nama	Role	Task
12S18014 Giovanni Situmorang	Data Analyst	Berperan dalam mengumpulkan, mengidentifikasi, menafsirkan serta menganalisis data, model, dan strategi yang efisien untuk digunakan dalam pengerjaan proyek.
	Programmer	Berperan dalam mengimplementasikan code untuk membangun sistem dan melakukan pengujian terhadap sistem yang sudah dibangun.
12S18015 Della Cenovita Tarigan	Data Analyst	Berperan dalam mengumpulkan, mengidentifikasi, menafsirkan serta menganalisis data, model, dan strategi yang efisien untuk digunakan dalam pengerjaan proyek.
	Programmer	Berperan dalam mengimplementasikan code untuk membangun sistem dan melakukan pengujian terhadap sistem yang sudah dibangun.
12S18022 Alex Conro Manuel	Data Analyst	Berperan dalam mengumpulkan, mengidentifikasi, menafsirkan serta menganalisis data, model, dan strategi yang efisien untuk digunakan dalam pengerjaan proyek.
	Programmer	Berperan dalam mengimplementasikan code untuk membangun sistem dan melakukan pengujian terhadap sistem yang sudah dibangun.
12S18042 Indah Oktavia M Sibarani	Data Analyst	Berperan dalam mengumpulkan, mengidentifikasi, menafsirkan serta menganalisis data, model, dan strategi yang efisien untuk digunakan dalam pengerjaan proyek.
	Programmer	Berperan dalam mengimplementasikan code untuk membangun sistem dan melakukan pengujian terhadap sistem yang sudah dibangun.
12S18051 Cindy Juniati	Data Analyst	Berperan dalam mengumpulkan, mengidentifikasi, menafsirkan serta menganalisis data, model, dan strategi

Hutapea		yang efisien untuk digunakan dalam pengerjaan proyek.
	Programmer	Berperan dalam mengimplementasikan code untuk membangun sistem dan melakukan pengujian terhadap sistem yang sudah dibangun.
12S18062 Ester Putri Dearest Sidabutar	Data Analyst	Berperan dalam mengumpulkan, mengidentifikasi, menafsirkan serta menganalisis data, model, dan strategi yang efisien untuk digunakan dalam pengerjaan proyek.
	Programmer	Berperan dalam mengimplementasikan code untuk membangun sistem dan melakukan pengujian terhadap sistem yang sudah dibangun.

3.2 Kesimpulan

Penamaan entitas pada bahasa Indonesia dengan menerapkan metode *Bidirectional Long Short-Term Memory* (BiLSTM) dan model ELMo dapat mengidentifikasi kata kunci serta mengklasifikasi ke dalam kategori tertentu. Dalam mengimplementasikan model, kelompok numpy belum berhasil menerapkan penggabungan antara BiLSTM model CRF dalam Analisis penamaan entitas pada bahasa Indonesia. Setelah dievaluasi penamaan entitas dalam bahasa Indonesia dengan pemodelan tersebut telah mendapatkan hasil akurasi sebesar 95,9%. Berdasarkan hasil confusion matrix dengan menggunakan test_label dan pred_label maka didapatkan nilai precision untuk Organisation sebesar 0.37, Person sebesar 0.34 dan Place sebesar 0.27, nilai recall untuk Organisation sebesar 0.14, Person sebesar 0.11 dan Place sebesar 0.40 dan nilai f1-score untuk Organisation sebesar 0.21, Person sebesar 0.17 dan Place sebesar 0.51. Berdasarkan hasil pemodelan tersebut, maka penamaan entitas pada bahasa indonesia dengan menggunakan BiLSTM dapat dikatakan akurat dengan hasil akurasi 95,9%.

3.3 Saran

Beberapa saran yang sekiranya dapat membantu meningkatkan efektivitas dari penamaan entitas dalam bahasa Indonesia sebagai berikut:

1. Penambahan jumlah data yang sekiranya dapat membantu dalam memperbanyak kosa kata dalam data training.
2. Membuat dataset yang digunakan dalam keadaan seimbang (balance) antara setiap kategori, seperti Person, Organisation, place, dan kategori lainnya.
3. Pemodelan dapat dilakukan dengan menggunakan algoritma yang lainnya untuk penamaan entitas dalam bahasa Indonesia yang lebih akurat, misalnya dengan menggunakan CRF, dll.

DAFTAR PUSTAKA

[1] SUFA, M., Yusliani, N., & Buchari, M. A. (2020). NAMED-ENTITY RECOGNITION PADA TEKS BERBAHASA INDONESIA MENGGUNAKAN METODE HIDDEN MARKOV MODEL DAN PART-OF-SPEECH TAGGING (Doctoral dissertation, Sriwijaya University).

[2] Aryoyudanta, B., Adji, T. B., & Hidayah, I. (2016, July). Semi-supervised learning approach for Indonesian Named Entity Recognition (NER) using co-training algorithm. In 2016 International Seminar on Intelligent Technology and Its Applications (ISITIA) (pp. 7-12). IEEE.

[3] Aditya Satrya Wibawa, Ayu Purwarianti. (2016). Indonesian Named-entity Recognition for 15 Classes Using Ensemble Supervised Learning.

[4] Zhang, M., Geng, G., & Chen, J. (2020). Semi-supervised bidirectional long short-term memory and conditional random fields model for named-entity recognition using embeddings from language models representations. Entropy, 22(2), 252.

[5] I. Budi and S. Bressan, "Application of association rules mining to Named Entity Recognition and co-reference resolution for the Indonesian language," International Journal of Business Intelligence and Data Mining, vol. 2, no. 4, p. 426–446, 2007.

[6] Gunawan, W., Suhartono, D., Purnomo, F., & Ongko, A. (2018). Named-entity recognition for indonesian language using bidirectional lstm-cnns. *Procedia Computer Science*, *135*, 425-432.

[7] Permana, H. (2019). *Named Entity Recognition Menggunakan Metode Bidirectional Lstm-Crf Pada Teks Bahasa Indonesia* (Doctoral dissertation, Universitas Komputer Indonesia).