

Dimas Aditya Faiz (1706039686)

Yaumi Alfadha (1706023031)

NER

a. Pilihlah dua1 dari 10 test\_set yang diberikan pada file test\_data. Anotasi label NER pada test\_set yang Anda pilih.

NER Test Set: 1, 2

NER Fitur: 1, 2, 3

b. Anda diminta mengimplementasikan kelompok fitur untuk model NER. Setiap kelompok diminta mengimplementasikan 3 kelompok fitur berikut2 (tidak termasuk bonus). Implementasi mencakup ekstraksi fitur terhadap training data dan testing data.

Kami memilih 3 kelompok fitur, antara lain :

- i. Token dan context windows
- ii. Karakter n-gram
- iii. Informasi case

code dan output snippet untuk fitur yang digunakan :

```
features = {
    'bias': 1.0,
    'word.lower()': word.lower(),
    'word[-3:]': word[-3:],
    'word[-2:]': word[-2:],
    'word.isupper()': word.isupper(),
    'word.istitle()': word.istitle(),
    'word.isdigit()': word.isdigit(),
    'postag': postag,
    'postag[:2]': postag[:2],
    '2_gram' : gram_2(word),
    '3_gram' : gram_3(word),
    '4_gram' : gram_4(word),
    'word.IsInitCaps()': IsInitCaps(word),
    'word.IsMixedCaps()': IsMixedCaps(word),
}
```

```
[251] # Split X and y into training (80%) and test (20%) sets
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

      print("First token features:\n{}\n{}".format("-"*21, X_train[0][0]))
      print("\nFirst token label:\n{}\n{}".format("-"*18, y_train[0][0]))

      First token features:
      -----
      {'bias': 1.0, 'word.lower()': 'jero', 'word[-3:]': 'ero', 'word[-2:]': 'ro', 'word.isupper()': False, 'word.istitle()': True, 'word.isdigit':
      False}

      First token label:
      -----
      B-PERSON
```

Bonus: mengimplementasikan fitur dengan memanfaatkan tugas NLP lainnya seperti POS tagging, word embedding, dan lain-lain.

Kami juga mengimplementasikan pos tagging pada train data set, sehingga train data yang kami gunakan adalah sebagai berikut:

```
[ ] post_tag([list(map(tuple, sentences[0]))])
```

	words	pos	ne
0	Agus	NNP	O
1	digantikan	VB	O
2	Gatot	NNP	O
3	Lupri	NNP	O
4	Jantomo	NNP	O
5	dari	IN	O
6	daerah	NN	O
7	pemilhan	NN	O
8	Jawa	NNP	O
9	Tengah	NNP	O
10	VIII	NNP	O

c. Lakukan pelatihan model CRF untuk melakukan NER dengan menggunakan data latih yang diberikan dan fitur yang telah diimplementasikan sebelumnya. Kemudian lakukan prediksi terhadap test\_set.

Output snippet untuk labelling :

```
test_1 = pd.DataFrame(
    {'Kalimat': los2[0],
     'Prediksi': set_pred[0],
    })
test_1.T
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Kalimat	Tengko	Amir	Hamzah	yang	bernama	lengkap	Tengko	Amir	Hamzah	Pangeran	Indra	Poetera	atau	lebih	dikenal	hanya
Prediksi	B- PERSON	I- PERSON	I- PERSON	O	O	O	B- PERSON	I- PERSON	I- PERSON	I- PERSON	I- PERSON	I- PERSON	O	O	O	O

```
[239] test_2 = pd.DataFrame(
    {'Kalimat': los2[2],
     'Prediksi': set_pred[2],
    })
test_2.T
```

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Kalimat	Bahkan	setelah	Amir	melanjutkan	studinya	di	sekolah	hukum	di	Batavia	(sekarang	Jakarta)	keduanya	tetap	dekat,	hanya
Prediksi	O	O	B- PERSON	O	O	O	O	O	O	B- LOCATION	I- LOCATION	I- LOCATION	O	O	O	O

d. Bandingkan hasil prediksi model dengan hasil anotasi. Evaluasi NER model dengan variasi pengukuran3 : token-level vs entity-level, exact match vs partial match, dan macro average vs micro average. Metrik yang diukur mencakup precision, recall, dan F1-score.

## exact match vs partial match

Measure	Type	Partial	Exact	Strict
Correct	281	281	281	281
Incorrect	8	8	8	8
Partial	0	0	0	0
Missed	0	0	0	0
Spurious	0	0	0	0
Precision	0.9723183391 00346	0.97231833910 0346	0.9723183391 00346	0.9723183391 00346
Recall	0.9723183391 00346	0.9723183391 00346	0.9723183391 00346	0.9723183391 00346
F1	0.9723183391 00346	0.9723183391 00346	0.9723183391 00346	0.9723183391 00346

## WSD

- A. Lakukan prediksi terhadap data testing dengan menggunakan model yang dibuat pada tahap sebelumnya

Snippet output (kata 'Lebat'):

```
[370] fin2 = test2_cut[["kata", "sense", "id"]]
      fin2['Prediction'] = test2_pred
      fin2
```

	kata	sense	id	Prediction
152	lebat	5501	932115.0	5501
153	lebat	5501	932135.0	5501
154	lebat	5502	932157.0	5501
155	lebat	5501	932158.0	5501
156	lebat	5502	932175.0	5502
157	lebat	5501	932185.0	5501
158	lebat	5501	932207.0	5501
159	lebat	5501	932229.0	5501
160	lebat	5501	932270.0	5501

✓ 0 d selesai pada 23.40

Snippet output (kata 'Mata'):

```
[378] fin6 = test6_cut[["kata", "sense", "id"]]
      fin6['Prediction'] = test6_pred
      fin6
```

```
17  mata    1003  946074.0    1003
18  mata    1001  946155.0    1001
19  mata    1003  946355.0    1003
20  mata    1001  946410.0    1001
21  mata    1002  946485.0    1002
22  mata    1003  946534.0    1003
23  mata    1006  946550.0    1001
24  mata    1001  946842.0    1001
25  mata    1001  946849.0    1001
26  mata    1006  946999.0    1001
```

B. accuracy, precision, recall, dan F1-score.

Kata 'lebat':

```
[141] y2 = fin2['sense'].values.tolist()
      accuracy_score(test2_pred, y2)*100
```

```
91.30434782608695
```

```
[142] print(classification_report(test2_pred,y2))
```

	precision	recall	f1-score	support
5501	1.00	0.89	0.94	18
5502	0.71	1.00	0.83	5
accuracy			0.91	23
macro avg	0.86	0.94	0.89	23
weighted avg	0.94	0.91	0.92	23

✓ 0 d selesai pada 00.25

Kata 'mata':

```
[171] y6 = fin6['sense'].values.tolist()
      accuracy_score(test6_pred, y6)*100
```

73.46938775510205

```
[172] print(classification_report(test6_pred,y6))
```

	precision	recall	f1-score	support
1001	1.00	0.55	0.71	29
1002	1.00	1.00	1.00	6
1003	0.88	1.00	0.93	14
1004	0.00	0.00	0.00	0
1005	0.00	0.00	0.00	0
1006	0.00	0.00	0.00	0
accuracy			0.73	49
macro avg	0.48	0.43	0.44	49
weighted avg	0.96	0.73	0.81	49

C. tidak menggunakan seluruh fitur(top 50)  
kata lebat :

```
y2 = fin2['sense'].values.tolist()
accuracy_score(test2_pred, y2)*100
```

69.56521739130434

```
print(classification_report(test2_pred,y2))
```

	precision	recall	f1-score	support
5501	1.00	0.70	0.82	23
5502	0.00	0.00	0.00	0
accuracy			0.70	23
macro avg	0.50	0.35	0.41	23
weighted avg	1.00	0.70	0.82	23

Kata 'mata' :

```
[192] y6 = fin6['sense'].values.tolist()
      accuracy_score(test6_pred, y6)*100
```

↳ 40.816326530612244

```
print(classification_report(test2_pred,y2))
```

	precision	recall	f1-score	support
1001	0.00	0.00	0.00	23.0
5501	0.00	0.00	0.00	0.0
5502	0.00	0.00	0.00	0.0
accuracy			0.00	23.0
macro avg	0.00	0.00	0.00	23.0
weighted avg	0.00	0.00	0.00	23.0

Akurasi yang didapatkan menurun jika jumlah train data dikurangi, Hal ini menunjukkan bahwa model tidak mendapatkan informasi yang cukup untuk melakukan labelling secara akurat.

#### Refrensi

- <https://datascience.stackexchange.com/questions/15989/micro-average-vs-macro-average-performance-in-a-multiclass-classification-settin>
- <https://github.com/farhanreynaldo/pos-tagging-indonesia>
- <https://towardsdatascience.com/named-entity-recognition-and-classification-with-scikit-learn-f05372f07ba2>
- <https://sklearn-crfsuite.readthedocs.io/en/latest/tutorial.html>