BAB V
UJI COBA

Bab kelima adalah uji coba yang dilakukan dalam tugas akhir ini. Dimulai dengan pembahasan cara hitung evaluasi dari Sequence-to-Set Network, yaitu Micro dan Macro dari F1 Score. Kemudian uji coba terhadap metode Sequence-to-Set Network. Uji coba akan dijelaskan dalam beberapa subbab bergantung pada konfigurasi yang diubah pada Sequence-to-Set Network. Kemudian akan dijelaskan metode pembanding dengan hasil Sequence-to-Set Network.

5.1 Evaluator Performa Hasil dari Tugas Akhir

Evaluator adalah cara penghitungan performa dari tugas akhir ini, tepatnya metode Sequence-to-Set Network dalam bahasa Indonesia. Performa prediksi sebuah metode biasanya dihitung dengan akurasi prediksi dengan target output/prediksi sebenarnya. Penghitungan ini akan menggunakan library/tool yang membantu penghitungan yang akurat dan benar. Akan ada dua jenis evaluator yaitu Micro F1 Score dan Macro F1 Score. Keduanya akan dijelaskan pada subbab masing-masing.

5.1.1 Macro dan Micro F1Score

Macro F1 Score adalah penghitungan F1 Score yang paling terus terang karena rumusnya. Macro F1 Score akan mengambil semua F1 Score yang telah dihitung, kemudian langsung dihitung nilai rata-rata tanpa ada perhitungan bobot lainnya. Contoh penghitungan akan diambil dari artikel "Micro, Macro & Weighted Averages of F1 Score, Clearly Explained"[1]. Disediakan hasil penghitungan True Positive (TP), False Positive (FP), and False Negative (FN) dari sebuah data klasifikasi gambar pesawat (dilambangkan A), kapal (dilambangkan B) dan mobil

[1] Kenneth Leung, Micro, Macro & Weighted Averages of F1 Score, Clearly Explained, (https://towardsdatascience.com/micro-macro-weighted-averages-of-f1-score-clearly-explained-b603420b292f#2f35)

(dilambangkan C) sebagai tabel berikut (Tabel 5.1). Tabel tersebut akan menyediakan metrik precision (P), recall (R), dan F1 score untuk tiap label.

Tabel 5.1
Contoh Data Confusion Matriks

Label	TP	FP	FN	Precision	Recall	F1 score
Pesawat	2	1	1	0,67	0,67	$\frac{2 \text{ x } (0{,}67 \text{ x } 0{,}67)}{(0{,}67 + 0{,}67)}$ = **0,67**
Kapal	1	3	0	0,25	1,00	$\frac{2 \text{ x } (0{,}25 \text{ x } 1{,}00)}{(0{,}25 + 1{,}00)}$ = **0,40**
Mobil	3	0	3	1,00	0,50	$\frac{2 \text{ x } (1{,}00 \text{ x } 0{,}50)}{(1{,}00 + 0{,}50)}$ = **0,67**

Penghitungan precision, recall dan F1 score dapat dilihat dari rumus $P = \frac{Jumlah\ semua\ TP}{Jumlah\ semua\ TP + Jumlah\ semua\ FP}$ (5.1, $R = \frac{Jumlah\ semua\ TP}{Jumlah\ semua\ TP + Jumlah\ semua\ FN}$ (5.2, dan $F1 = \frac{2*(P+R)}{(P\ x\ R)}$ (5.3. Precision dan recall tidak dapat digunakan sebagai evaluasi suatu performa model karena itu nilai tersebut akan digunakan untuk membantu menghitung F1 Score. F1 Score adalah penghitungan yang dibuat agar dapat melihat precision dan recall seimbang dan penghitungan F1 Score terbukti nilai evaluasi yang bagus (meskipun data mungkin tidak seimbang). Dengan rumus F1 Score yang di tunjukkan pada rumus $F1 = \frac{2*(P+R)}{(P\ x\ R)}$ (5.3, membuktikan jika model mendapat nilai precision dan recall yang tinggi makan nilai F1 Score, begitupun untuk nilai rendah. Jika model memiliki nilai precision dan recall yang salah satunya nilai rendah dan salah satunya lagi nilainya tinggi, akan menghasilkan F1 Score yang rata-rata.

$$P = \frac{Jumlah\ semua\ TP}{Jumlah\ semua\ TP + Jumlah\ semua\ FP} \quad \text{.....(5.1)}$$

$$R = \frac{Jumlah\ semua\ TP}{Jumlah\ semua\ TP + Jumlah\ semua\ FN} \quad \text{.....(5.2)}$$

$$F1 = \frac{2*(P+R)}{(P\ x\ R)} \quad \text{.....(5.3)}$$

Untuk Macro F1 Score, pada Tabel 5.2 menjelaskan bahwa seluruh F1 Score yang telah dihitung sebelumnya akan dijumlah dan dibagi sesuai jumlah label. Dengan kata lain, Macro F1 Score adalah penghitungan F1 Score rata-rata yang tidak berbobot. Artinya bahwa Macro F1 Score menganggap tiap label semua rata tanpa melihat jumlah *support* (jumlah kemunculan label dalam dataset) tiap label.

Tabel 5.2
Penghitungan Micro dan Macro

Label	TP	FP	FN	F1 score	Macro	Micro
Pesawat	2	1	1	0,67	$\frac{0,67 + 0,40 + 0,67}{3}$ = **0,58**	$\frac{\sum TP}{\sum TP + \frac{1}{2}(\sum FP + \sum FN)}$ $= \frac{6}{6 + \frac{1}{2}(4+4)}$ = **0,60**
Kapal	1	3	0	0,40		
Mobil	3	0	3	0,67		

Sedangkan Micro F1 Score akan menghitung nilai rata-rata global. Penghitungan menggunakan True Positive (TP), False Negative (FN), and False Positive (FP) dari semua data. Contoh penghitungan akan diberikan pada Tabel 5.2 dan rumus dapat dilihat pada dalam tabel tersebut . Tiap TP, FN dan FP dari seluruh label akan dijumlah dan digunakan dalam rumus Micro F1 Score. Micro F1 Score dapat juga dibilang akurasi (*accuracy*), karena pada dasarnya menghitung proporsi prediksi yang tepat dari semua prediksi. Dengan begitu, definisi tesrebut yang kita gunakan untuk menghitung akurasi secara keseluruhan.

Penggunaan Micro dan Macro perlu diperhatikan, untuk dataset yang tidak seimbang datanya namun tiap jenis data/label/*class* sejajar kepentingannya, maka dapat menggunakan Macro F1 Score. Jika dataset yang dimiliki dianggap cukup seimbang dan ingin nilai metrik yang dapat menyimpulkan performa secara keseluruhan label, lebih baik menggunakan Micro F1 Score. Pada Sequence-to-Set

Network jenis F1 Score yang digunakan adalah Macro F1 Score dan Micro F1 Score. Namun untuk perbandingan F1 Score terbaik diambil dari Micro F1 Score.

Seperti yang telah disebut, penghitungan dalam program tugas akhir ini menggunakan bantuan library. Library tersebut adalah *sklearn*, secara khusus fungsi yang dibutuhkan adalah precision_recall_fscore_support dari modul *metrics*. Dengan menggunakan fungsi tersebut, dengan singkat penulisan program langsung mendapat nilai precision, recall, F1 Score, bahkan juga support tiap label. Parameter dari fungsi tersebut yang digunakan adalah y_true dan y_pred diisi dengan golden entites dan entitas yang telah diprediksikan. Kemudian dua parameter lainnya adalah *labels* untuk memberikan jenis class/label yang ada, dalam tugas akhir ini label yang digunakan adalah jenis-jenis entitas. Parameter terakhir yang digunakan adalah *average*, yang digunakan adalah micro dan macro namun selain itu ada beberapa jenis average yang disediakan sklearn seperti *weighted*, *samples*, *binary*.

5.2 Uji Coba Sequence-to-Set Network pada Nested NER

Konfigurasi dari *hyperparameter* yang akan diubah untuk uji coba ini ada empat jenis yaitu *Batch Size*, *Learning Rate*, *Gradient Norm*, *Dropout*. Penilaian akurasi menggunakan Micro F1 Score dan diambil yang terbaik dari seluruh *epoch* yang dilewatkan. Pembagian dataset adalah 90% *training* dan *development* dan 10% untuk *testing*. Total ada 42 hyperparameter yang dapat dikonfigurasikan, namun akan disebut hyperparamter yang akan diubah untuk uji coba.

Nilai *default* untuk tiap hyperparameter adalah batch size sejumlah delapan, learning rate dengan nilai 2e-5, gradient norm dengan nilai 1.0 dan nilai dropout 0.1. Catatan untuk hyperparamter dropout, terdapat tiga hyperparameter yang dapat diubah nilai dropout nya yaitu prop_drop, lstm_drop dan char_lstm_drop. Hyperparameter tersebut masing-masing yaitu, prop_drop untuk merubah nilai probabilitas dropout dalam training model Sequence-to-Set Network. lstm_drop untuk merubah nilai probabilitas dropout untuk embedding akhir BiLSTM. char_lstm_drop untuk merubah nilai probabilitas dropout untuk embedding character-level BiLSTM. Uji coba dilakukan dengan cara untuk hyperparameter

yang sedang diuji coba akan diganti, sedangkan hyperparameter lainnya mengikuti nilai default.

Untuk tiap gambar Secara kesimpulan dari uji coba Sequence-to-Set Network adalah F1 Score tertinggi adalah uji coba perubahan untuk batch size sebanyak dua dengan F1 Score 72.28%. Detail hasil uji coba untuk tiap jenis entitas, baik F1 Score, support, precision dan recall telah ditampilkan pada Tabel 5.3

Tabel 5.3
Hasil Uji Coba F1 Score Terbaik

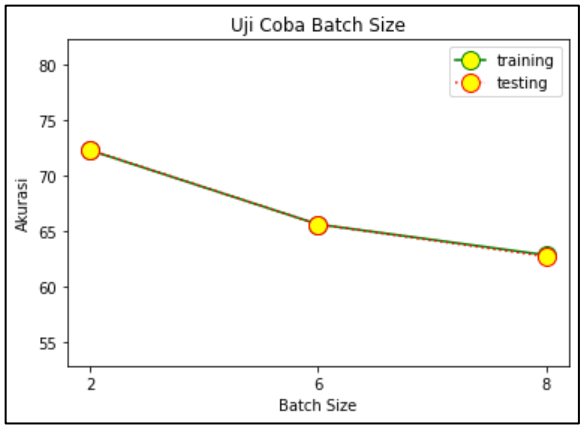
Jenis Entitas	Precision	Recall	F1 score	Support
TIME	67.21	75.21	70.98	714
LOC	71.33	70.88	71.10	1583
EVENT	67.73	48.53	56.55	307
DATE	88.87	91.36	90.10	463
PER	75.62	81.10	78.26	3212
MISC	49.27	56.16	52.49	479
ORG	67.10	68.04	67.57	2353
MICRO	71.00	73.60	72.28	9111
MACRO	69.59	70.18	69.58	9111

5.2.1 Pengaruh Batch Size

Uji coba pertama adalah perubahan hyperparameter jumlah batch size terhadap nilai akurasi terbaik diakhir uji coba. Nilai untuk batch size yang di uji coba adalah batch size sejumlah 2, batch size sejumlah 6, batch size sejumlah 8 (nilai default). Hyperparameter lainnya akan mengikuti nilai default selama uji coba ini dilakukan. Gambar 5.1 menunjukkan hasil dari uji coba dengan merubah jumlah batch size.

Dapat dilihat perubahan terjadi cukup signifikan untuk batch size jumlah dua menuju batch size jumlah 6, sebanyak 7% (72.28 dan 65.59). Namun untuk perubahan dari batch size jumlah 6 menuju batch size jumlah 8 hanya sebanyak sekitar 3% (62.82 dan 65.59). Untuk akurasi saat proses testing, hasil akurasi tidak

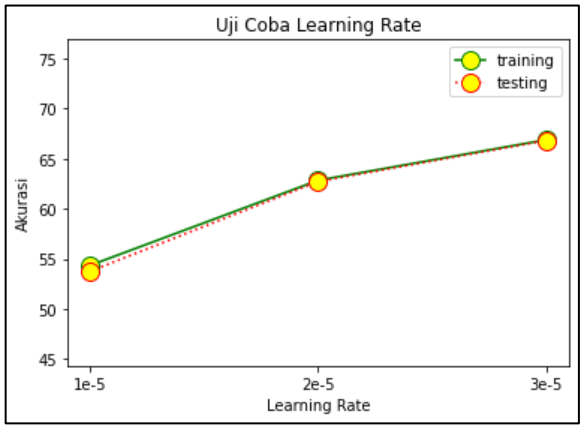
memiliki perbedaan jauh, perbedaan ditemukan kurang dari satu (72.31, 65.59, dan 62.7). Kesimpulan sementara untuk pengaruh jumlah batch size adalah mempengaruh tidak terlalu signifikan (kurang lebih 3%). Ada pun kesimpulan berkurangnya nilai F1 Score semakin bertambah jumlah batch size.



Gambar 5.1
Hasil Uji Coba Jumlah Batch Size

5.2.2 Pengaruh Learning Rate

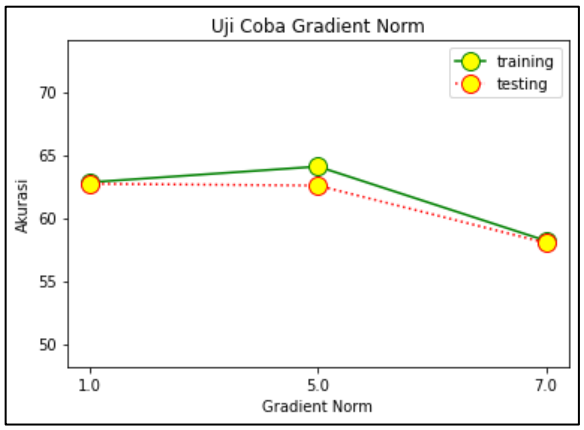
Uji coba berikutnya perubahan hyperparameter nilai learning rate. Nilai untuk learning rate yang di uji coba adalah 1e-5, 2e-5 (nilai default) dan 3e-5. Selain learning rate akan bernilai default selama uji coba ini dilakukan. Gambar 5.2 menunjukkan hasil dari uji coba dengan merubah learning rate. Perubahan yang muncul nilai learning rate 1e-5 menuju learning rate 2e-5, sebanyak 4% (54.34 dan 62.82). Jarak perubahan F1 Score dari uji coba kedua dan ketiga juga sama dengan sebelumnya yaitu 4% (62.82 dan 66.86). Akurasi proses testing tidak berbeda jauh (53.75, 62.7, dan 66.79). Kesimpulan dari uji coba ini tidak terlalu signifikan dan makin bertambah nilainya makin tinggi nilai F1 Score.



Gambar 5.2
Hasil Uji Coba Learning Rate

5.2.3 Pengaruh Gradient Norm

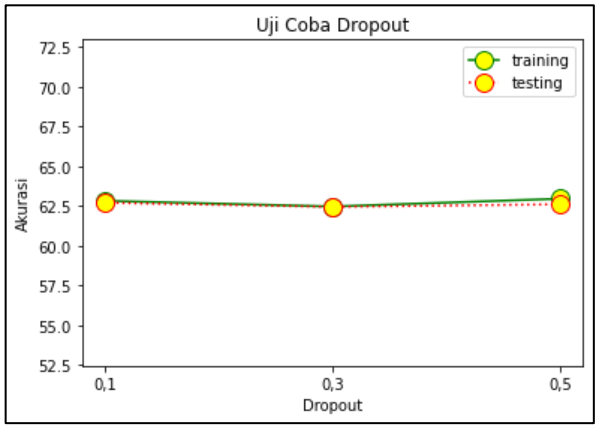
Uji coba berikutnya adalah perubahan nilai gradient norm. Nilai yang akan di uji coba adalah 1.0 (nilai default), 5.0, 7.0 dan hyperparameter lainnya mengikuti nilai default pada uji coba ini. Gambar 5.3 menunjukkan hasil dari uji coba dengan perubahan yang muncul adalah dari nilai 1,0 menuju 5,0, terjadi sekitar 2% (62.82 dan 64.07). Tetapi saat nilai dari gradient norm dinaikkan, akurasi mengalami penuruan sebanyak 7% sehingga dapat dikatakan perubahan jika gradient norm dinaikkan mengalami perubaha signifikan namun penuruan.



Gambar 5.3
Hasil Uji Coba Gradient Norm

5.2.4 Pengaruh Dropout

Uji coba terakhir adalah perubahan hyperparameter nilai dropout yang akan dilakukan kepada tiga variabel berbeda seperti yang dijelaskan (prop_drop, lstm_drop dan char_lstm_drop). Nilai drouput yang di uji coba adalah nilai 0.1 (nilai default), 0.3, 0.5. Hyperparameter lainnya akan mengikuti nilai default selama uji coba ini dilakukan. Gambar 5.4 menunjukkan hasil dari uji coba perubahan nilai dropout dan hasil dari perubahannya tidak signifikan dengan perubahan F1 Score yang tidak lebih dari 1% (akurasi training adalah 62.82, 62.46, 62.95). Perbandingan dengan akurasi test juga tidak beda jauh (62.7, 62.44, dan 62.6). Kesimpulan untuk pengaruh perubahan ketiga variabel dropout adalah mempengaruh tidak signifikan (kurang dari 1%).



Gambar 5.4
Hasil Uji Coba Dropout

5.3 Perbandingan Metode Span-Based Method

Nested NER dari sebelumnya memiliki metode yang dianggap sudah cocok untuk digunakan sebagai penelitian, dan metode ini sudah berkembang banyak dalam Nested NER. Metode ini menganggap pengenalan entitas dengan klasifikasi span (rentangan, suatu bagian dengan batasan kiri dan kanan). Tetapi kekurangan dari metode span-based ini dengan konsep span, *search space* (luas pencarian) menjadi terlalu besar dan metode span-based tidak melihat nilai konteks satu entitas dengan entitas lain, yang sebenarnya dapat membantu model untuk menentukan entitas berdasarkan konteksnya. Karena itu, adanya Sequence-to-Set Network

untuk memberikan solusi komputasi yang tidak seberat metode span-based tetapi memberikan akurasi yang tinggi.

Perbandingan dengan metode ini tidak dengan dataset tugas akhir ini namun dengan dataset GENIA. Alasannya karena metode pembanding membutuhkan komputasi yang besar dan resource dari tugas akhir ini tidak memadainya. Kedua, dengan dataset GENIA dan word embedding/model pretrained yang berhubungan dengan kata-kata biologi akan lebih stabil sebagai perbandingan.

Metode yang akan dibandingkan adalah metode Locate and Label[2]. Locate and Label adalah metode span-based terbaru pada saat tugas akhir ini dikerjakan. Tidak hanya sebagai metode span-based/klasifikasi span, metode ini menggunakan *identifier* entitas yang memiliki dua tahap. Pertama, akan diusulkan span dengan filter dan *boundary regression* pada *seed span* untuk menemukan entitas, kemudian diberikan label kepada span yang diusulkan dan disesuaikan batas dengan kategori yang sesuai. Perbandingan F1 Score untuk metode Sequence-to-Set dengan Locate and Label dapat dilihat dari Tabel 5.4. Dari tabel tersebut dapat dilihat F1 Score yang dimiliki keduanya hanya berbeda 0,10%. Meskipun begitu, Locate and Label menang dengan persentase 80,54% sedangkan Sequence-to-Set dengan 80,44%. Keunggulan dari Locate and Label adalah keberhasilan metode tersebut dalam metrik Recall yang menjadi 80,89% sedangkan Sequence-to-Set dengan 78,66%. Yang unggul Sequence-to-Set adalah arsitektur sistem yang lebih hemat dibandingkan dengan Locate and Label.

Tabel 5.4
Tabel Perbandingan F1 Score

Model	Precision	Recall	F1 Score
Lin et al. (2019)	75.80	73.90	74.80
Luo and Zhao (2020)	77.40	74.60	76.00
Wang et al. (2020b)	78.10	74.40	76.20

[2] Yongliang Shen, Locate and Label: A Two-stage Identifier for Nested Named Entity Recognition, Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), (Agustus : 2021).

Strakova et al. (2019)	-	-	78.31
Wang et al. (2020a)	79.45	78.94	79.19
Yu et al. (2020)	81.80	79.30	80.50
Sequence-to-Set (2021)	82.31	78.66	80.44
Locate and Label (2021)	80.10	80.89	80.54

5.4 Catatan Uji Coba

Pelaksanaan uji coba tugas akhir ini memiliki beberapa kesulitan, kesulitan utamanya adalah kurangnya *resource* (fasilitas/sumber daya) secara memory untuk menjalankan arsitektur Sequence-to-Set secara keseluruhan. Penelitian pada awal dijalankan pada *environment* OS Linux (dapat dilihat dalam Lampiran B) dengan GPU GeForce Nvidia RTX 3070 8GB. Karena tidak memadai, arsitektur diupayakan di kecilkan dengan cara Autocast[3] yaitu hidden size tidak dikurangi namun beberapa dari weight akan dibuang untuk meringankan komputasi, cara ini tidak dapat memenuhi kebutuhan dari arsitektur. Kemudian hidden size dari Transformer dan FFN dikurangi dari nilai awal 782 dan 1028 menjadi rata 504 untuk tiap dimensi. GPU dapat menjalankan metode namun mendapatkan akurasi yang kurang bagus karena pemotongan hidden size yang sekitar 50%.

Dengan ini, diputuskan membutuhkan resource yang lebih besar, sehinga penelitian menggunakan Google Colab Pro yang menyediakan GPU Nvidia Tesla T4 16GB. Tetapi GPU tersebut masih belum bisa memenuhi Transformers, sehingga penelitian dilanjutkan dengan Transformers tidak di training ulang (freeze), namun Autocast tidak digunakan, dan hidden size disesuaikan kembali 782 dan 1028. Ada pun beberapa halangan dalam Google Colab Pro karena beberapa fitur yang tidak dapat diakses. Idle runtime diberikan secara random, sehingga notebook tidak dapat ditinggalkan sepenuhnya, harus dipantau agar tidak terputus koneksi secara mendadak (unexpected error, dapat dilihat pada lampiran B). Ditengah pelaksanaan uji coba juga terjadi runtime disconnected, yang berarti Google Colab tidak mengijinkan komputasi berat tanpa aktifitas yang interaktif

[3] Automatic Mixed Precision Package, (https://pytorch.org/docs/stable/amp.html).

(terlalu idle). Karena penggunaan GPU akun tugas akhir ini terlalu berat dan tidak interaktif, maka Google Colab memberikan timeout selama 5 jam[4].

[4] Google Colab FAQ, https://research.google.com/colaboratory/faq.html