KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* BERDASARKAN DNA *SEQUENCE* MENGGUNAKAN METODE *GATED RECURRENT UNIT* (GRU)

(Skripsi)

Oleh:

AZAHRA ALYA HIDAYAH
1917051017

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2023

ABSTRAK

KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* BERDASARKAN DNA *SEQUENCE* MENGGUNAKAN METODE *GATED RECURRENT UNIT* (GRU)

Oleh

Azahra Alya Hidayah

Gen esensial merupakan gen yang sangat penting dan diperlukan untuk kelangsungan hidup suatu organisme. Informasi terkait esensialitas gen digunakan dalam berbagai penelitian ilmu *science*. Gen esensial dapat diidentifikasi melalui DNA. Gen-gen ini dapat diidenfitikasi secara eksperimental dengan berbagai teknik, namun memerlukan sumber daya yang besar. Untuk mengatasi hal tersebut, metode komputasi digunakan untuk mengklasifikasikan gen esensial. Metode yang digunakan adalah *Gated Recurrent Unit* (GRU). Pada penelitian ini menggunakan *Drosophila melanogaster* yang merupakan salah satu organisme yang sering dijadikan model dalam penelitian *science*. Tujuan dari penelitian ini adalah untuk mengetahui kinerja metode *Gated Recurrent Unit* (GRU) dalam mengklasifikasikan DNA *sequence* pada *Drosophila melanogaster*. Dataset yang digunakan diperoleh dari penelitian Beder, et al, (2021) dengan 2 jenis dataset yaitu *Cellular Essential Gene* (CEG) dan *Organismal Essential Gene* (OEG). Terdapat tiga skema arsitektur model dan dua skenario pembagian data, yaitu 80% *training* 20% *validation* dan 90% *training* 10% data *validation.* Pada dataset CEG memiliki distribusi kelas yang tidak seimbang sehingga dilakukan proses *Random Undersampling* (RUS) untuk menyeimbangkan kelas. Hasil kinerja yang paling baik pada dataset OEG didapatkan pada skenario pembagian data 90% *training* 10% *validation* dengan nilai yang didapat adalah 73 % *sensitivity*, 72% *specificity*, 73% nilai ROC-AUC dan 78% nilai PR-AUC. Hasil yang paling baik pada dataset CEG diperoleh dari pembagian data 90% *training* 10% *validation* dengan nilai yang didapat untuk *sensitivity* adalah 72%, *specificity* 48%, nilai ROC-AUC 60% dan nilai PR-AUC 44%.

Kata Kunci : Gen Esensial, *Drosophila melanogaster*, DNA *sequence*, GRU.

ABSTRACT

CLASSIFICATION OF ESSENTIAL GENES IN DROSOPHILA MELANOGASTER BASED ON DNA SEQUENCE USING GATED RECURRENT UNIT (GRU)

By

Azahra Alya Hidayah

Essential genes are genes that are very important and necessary for the survival of an organism. Information about essentiality is used in various scientific research studies. Essential genes can be identified through DNA. Genes can be identified experimentally using various techniques, but it requires substantial resources. To solve this problem, computational methods are used to classify essential genes. The used method is Gated Recurrent Unit (GRU). In this research, Drosophila melanogaster was used as the research object. Drosophila melanogaster is an organism that is often used as an object in scientific research. The purpose of this research is to determine the performance of the GRU method in classifying Drosophila melanogaster's DNA sequences. The dataset was obtained from research by Beder, et al, (2021) with 2 types of dataset, namely Cellular Essential Gene (CEG) and Organismal Essential Gene (OEG). There are three model architecture schemes and two data separation scenarios, 80% training 20% validation and 90% training 10% validation. The CEG dataset has an imbalanced class distribution so a Random Undersampling (RUS) process is carried out to balance the classes. The best performance results on OEG dataset obtained in data separation 90% training 10% validation scenario, the result are 73% sensitivity, 72% specificity, 73% ROC-AUC and 78% PR-AUC. The best results on CEG dataset obtained in data separation 90% training 10% validation with the values are 72% sensitivity, 48% specificity, 60% ROC-AUC and 44% PR-AUC.

Keywords : Essential genes, Drosophila melanogaster, DNA sequence, GRU.

KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* BERDASARKAN DNA *SEQUENCE* MENGGUNAKAN METODE *GATED RECURRENT UNIT* (GRU)

Oleh:

AZAHRA ALYA HIDAYAH

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar
SARJANA ILMU KOMPUTER

Pada

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Lampung

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2023

Judul Skripsi	:	**KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* BERDASARKAN DNA *SEQUENCE* MENGGUNAKAN METODE *GATED RECURRENT UNIT* (GRU)**
Nama Mahasiswa	:	Azahra Alya Hidayah
Nomor Pokok Mahasiswa	:	1917051017
Program Studi	:	Ilmu Komputer
Fakultas	:	Matematika dan Ilmu Pengetahuan Alam

MENYETUJUI

1. Komisi Pembimbing

Favorisen R. Lumbanraja, Ph.D.
NIP 19830110 200812 1 002

2. Ketua Jurusan Ilmu Komputer

Didik Kurniawan, S.Si., M.T.
NIP 19800419 200501 1 004

MENGESAHKAN

1. Tim Penguji

   Ketua : **Favorisen R. Lumbanraja, Ph.D.** .....

   Penguji I
   Penguji Pembahas : **Fatma Indriani, S.T., MIT, Ph.D.** .....

   Penguji II
   Penguji Pembahas : **Dr. rer. nat Akmal Junaidi, M.Sc.** .....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr. Eng. Heri Satria, S.Si., M.Si.
NIP 19711001 200501 1 002

Tanggal Lulus Ujian Skripsi : **22 November 2023**

PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Azahra Alya Hidayah

NPM : 1917051017

Dengan ini menyatakan bahwa skripsi saya yang berjudul "KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* BERDASARKAN DNA *SEQUENCE* MENGGUNAKAN METODE *GATED RECURRENT UNIT* (GRU)" adalah benar hasil karya sendiri dan bukan orang lain. Seluruh tulisan yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Jika di kemudian hari terbukti skripsi saya adalah hasil penjiblakan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan peraturan yang berlaku.

Bandar Lampung, 5 Desember 2023

Penulis

Azahra Alya Hidayah

NPM 1917051017

RIWAYAT HIDUP

Penulis dilahirkan di Desa Kalirejo, Kecamatan Kalirejo, Kabupaten Lampung Tengah, Lampung pada tanggal 25 Juli 2002, sebagai anak pertama dari dua bersaudara dari pasangan Bapak Pardimin dan Ibu Apriyani. Penulis menyelesaikan pendidikan pertama di TK Al-Ihya Kalirejo tahun 2008, kemudian melanjutkan pendidikan dasar di SDN 2 Kaliwungu dan selesai pada tahun 2014. Menempuh pendidikan menengah pertama di SMPN 1 Kalirejo yang diselesaikan pada tahun 2017. Kemudian melanjutkan pendidikan menengah atas di SMAN 1 Gadingrejo yang diselesaikan pada tahun 2019.

Penulis terdaftar sebagai mahasiswa Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung pada tahun 2019 melalui jalur Seleksi Nasional Masuk Perguruan Tinggi Negeri (SNMPTN). Selama menjadi mahasiswa, penulis melakukan beberapa kegiatan antara lain.

1. Menjadi anggota Adapter Himpunan Mahasiswa Jurusan Ilmu Komputer pada periode 2019/2020.
2. Menjadi anggota pengurus di Badan Khusus Himpunan Mahasiswa Jurusan Ilmu Komputer (Himakom) pada periode 2018/2019
3. Menjadi Asisten Dosen Jurusan Ilmu Komputer untuk mata kuliah Matematika pada periode semester ganjil tahun ajaran 2020/2021.
4. Menjadi Asisten Dosen Jurusan Ilmu Komputer untuk mata kuliah Struktur Data dan Algoritma pada periode semester genap tahun ajaran 2020/2021.
5. Menjadi Asisten Dosen Jurusan Ilmu Komputer untuk mata kuliah Basis Data pada periode semester ganjil tahun ajaran 2021/2022.

6. Menjadi Asisten Dosen Jurusan Biologi Terapan untuk mata kuliah Dasar-Dasar Bioinformatika pada periode semester genap tahun ajaran 2022/2023.
7. Melaksanakan Kerja Praktik di BPS Kabupaten Pringsewu pada tahun 2022.
8. Melaksanakan Kuliah Kerja Nyata (KKN) pada tahun 2022 di Desa Sumbermulyo, Kecamatan Sumberejo, Kabupaten Tanggamus, Lampung.

MOTTO

اِنَّ اللهَ لَا يُغَيِّرُ مَا بِقَوْ مٍ حَتّٰى يُغَيِّرُوْا مَا بِاَ نْفُسِهِمْ

“Sesungguhnya Allah tidak akan mengubah keadaan suatu kaum sebelum mereka mengubah keadaan diri mereka sendiri.”

(Ar-Ra’d : 11)

فَاِ نَّ مَعَ الْعُسْرِ يُسْرًا

“Maka sesungguhnya bersama kesulitan ada kemudahan”

(Al-Insyirah : 5)

“Setiap fase yang kamu jalani harus bisa mendatangkan pelajaran untuk naik ke fase berikutnya.”

(Merry Riana)

PERSEMBAHAN

Alhamdulillahirobbilalamin

Puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga dapat menyelesaikan skripsi ini. Sholawat serta salam saya sanjungkan kepada Nabi Muhammad SAW.

Aku persembahkan karya ini kepada :

Ayah dan Mama tersayang sebagai tanda terimaku. Terima kasih telah mendidik, membesarkanku dengan penuh kasih sayang. Terima kasih selalu mendoakan, memberikan semangat, dukungan, dan memberikan hal yang terbaik untukku. Terima kasih atas perjuangan dan pengorbanan kalian. Terima kasih Ayah dan Mama. Teruntuk Adikku, Almira dan seluruh keluarga besar terima kasih telah memberikan doa, semangat dan dukungan.

Sahabat, dan teman-teman yang selalu memberikan semangat, dukungan dan doa.

Almamater Tercinta, Universitas Lampung

SANWACANA

Puji syukur kehadirat Allah SWT yang telah memberikan berkah, rahmat dan hidayah-Nya, serta sholawat dan salam yang selalu tercurah kepada Nabi Muhammad SAW, sehingga penulis dapat menyelesaikan skripsi yang berjudul "Klasifikasi Gen Esensial Pada *Drosophila melanogaster* Berdasarkan DNA *Sequence* Menggunakan Metode *Gated Recurrent Unit* (GRU)" dengan baik.

Selama proses pengerjaan dan penulisan skripsi ini tidak terlepas dari dukungan banyak pihak, yang telah membimbing, membantu dan memberikan semangat, sehingga pada kesempatan ini penulis ingin menyampaikan ungkapan terima kasih kepada :

1. Ayah, mama, adik dan keluarga yang selalu mendoakan, menyemangati, memberi motivasi, dukungan serta kasih sayang yang tiada henti.
2. Bapak Favorisen R. Lumbanraja, Ph. D. selaku dosen pembimbing utama yang selalu membimbing, memberikan arahan, masukan dan saran sehingga skripsi ini dapat diselesaikan dengan baik
3. Ibu Fatma Indriani, S.T., M.I.T., Ph.D. selaku dosen pembahas pertama yang selalu memberikan masukan dan saran sehingga skripsi ini dapat diselesaikan dengan baik.
4. Bapak Dr. rer. nat. Akmal Junaidi, M. Sc. sebagai pembahas kedua yang selalu memberikan saran dan masukan sehingga penulisan skripsi ini dapat diselesaikan dengan baik.
5. Ibu Yunda Heningtyas, M.Kom. selaku dosen pembimbing akademik, yang telah membimbing selama perkuliahan di Jurusan Ilmu Komputer.
6. Bapak Dr. Eng. Heri Satria, S.Si., M.Si., selaku Dekan FMIPA Universitas Lampung.

7. Bapak Didik Kurniawan, S.Si., M.T., selaku Ketua Jurusan Ilmu Komputer Universitas Lampung.
8. Ibu Anie Rose Irawati, S.T., M.Cs., selaku sekretaris Jurusan Ilmu Komputer Universitas Lampung.
9. Bapak dan Ibu Dosen Jurusan Ilmu Komputer yang telah memberikan ilmu serta pengalaman hidup selama perkuliahan.
10. Ibu Ade Nora Maela, Bang Zainuddin, dan Mas Nofal yang telah membantu dalam segala urusan administrasi di Jurusan Ilmu Komputer.
11. Sahabat dan teman seperbimbingan, Jihan Cahya Fatimah dan Dina Putri Aulia yang selalu mewarnai di setiap proses perkuliahan, yang selalu ada saat suka maupun duka yang selalu menyemangati, memberikan pengalaman hidup, dan menguatkan. Ardella Dean Awalia dan Mohammad Fajar yang saling menguatkan, membantu, dan menyemangati satu sama lain.
12. Teman serumah : Takhfa Nur Asyifa, Anastasya Dian Nurratri, Dina Putri Aulia, dan Salwa Rizki yang selalu membantu, menyemangati dan menguatkan.
13. Teman-teman Jurusan Ilmu Komputer FMIPA Universitas Lampung angkatan 2019 yang telah memberikan cerita dalam masa perkuliahan.
14. Semua pihak yang telah berpartisipasi baik secara langsung maupun tidak langsung dalam membantu penyusunan skripsi ini.

Penulis menyadari, tentu banyak kekurangan dalam penyusunan skripsi ini yang disebabkan oleh keterbatasan pengetahuan, kemampuan serta pengalaman. Oleh karena itu, masukan dan saran sangat diharapkan sebagai evaluasi dan pembelajaran kedepannya. Semoga skripsi ini dapat menambah ilmu dan bermanfaat bagi semua pihak.

Bandar Lampung, 5 Desember 2023

Azahra Alya Hidayah

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel Halaman

DAFTAR GAMBAR

Gambar Halaman

DAFTAR KODE PROGRAM

Kode Program Halaman

I. PENDAHULUAN

1.1. Latar Belakang

Gen esensial didefinisikan sebagai gen yang diperlukan untuk kelangsungan hidup suatu organisme atau sel. Ketika gen esensial dihapus, maka dapat menyebabkan adanya mutasi bahkan kematian pada organisme. Hal tersebut menunjukkan bahwa gen esensial melakukan fungsi biologis yang penting. Menurut Zhang & Ren (2015) gen esensial memiliki kepentingan khusus, tidak hanya untuk fungsi biologis esensial mereka, tetapi juga dalam aplikasi praktis, seperti mengidentifikasi target obat yang efektif untuk bakteri dan jamur patogen.

Karakterisasi gen esensial penting untuk dilakukan. Dengan mengidentifikasi gen esensial dapat membantu mendapatkan pemahaman tentang mekanisme seluler dan molekul dasar yang menopang kehidupan (Campos, et al., 2020). Dalam penelitian *science*, gen esensial biasa digunakan untuk mengindentifikasi target obat, misalnya pada kanker. Memprediksi gen esensial tidak hanya untuk mengidentifikasi target obat untuk pengembangan antibodi dan vaksin tetapi juga berkontribusi terhadap industri, mikrobiologi makanan dan bioremediasi (Nandi, Gangguli & Sarkar, 2020).

Identifikasi gen esensial secara eksperimental seperti teknik *genetic foot-printing* (Gerdes, et al., 2003)*, gene knockouts* (Baba, et al., 2006), RNA *interference* (RNAi) (Agrawal, et al., 2003), dan *transposon* (Reznikoff & Winterberg, 2008) biasanya membutuhkan waktu yang lama dan biaya yang mahal (Nandi, Gangguli & Sarkar, 2020).

Untuk itu identifikasi dilakukan dengan metode pembelajaran mesin yang menyediakan algoritma komputasi untuk memudahkan identifikasi. Metode pembelajaran mesin memberikan keuntungan bahwa model-model yang dibuat mampu mempelajari pola-pola dari sejumlah data yang diberikan.

Drosophila melanogaster menjadi salah satu organisme yang banyak digunakan dalam penelitian dan paling intensif untuk dipelajari (Adams, et al., 2000). *Drosophila melanogaster* digunakan sebagai model untuk mempelajari berbagai ilmu *science*. Sekitar 75% dari semua gen penyakit manusia yang diketahui memiliki homolog dengan lalat buah (*Drosophila melanogaster)* (Ethan, 2005). Dalam identifikasi gen esensial *Drosophila melanogaster* juga digunakan sebagai model pembelajaran mesin, seperti pada penelitian yang dilakukukan oleh Aromolaran, et al., (2020) *Drosophila melanogaster* dijadikan objek untuk identifikasi gen esensial dengan metode *machine learning*, dimana pada penelitian tersebut didapatkan hasil yang terbaik pada *Xtream Gradient Boosting*. Selain itu, penelitian oleh Campos T, et al.,(2020) juga menggunakan *Drosophila melanogaster* sebagai objek untuk identifikasi gen esensial dengan hasil terbaik pada metode *Gradient Boosting Machine.*

Mempertimbangkan pentingnya mengidentifikasi gen esensial, pada penelitian ini menggunakan metode *Gated Recurrent Unit* (GRU) untuk mengidentifikasi gen esensial pada *Drosophila melanogaster* berdasarkan DNA *sequence*. *Gated Recurrent Unit* (GRU) adalah salah satu varian dari *Recurrent Neural Network* (RNN) yang merupakan metode yang dirancang untuk pemodelan dan prediksi data sekuens, sehingga metode ini dinilai dapat melakukan klasifikasi terhadap data DNA.

1.2. Rumusan Masalah

Berdasarkan pemaparan latar belakang, adapun rumusan masalah pada penelitian ini adalah :

1. Apakah metode *Gated Recurrent Unit* (GRU) dapat diimplementasikan untuk membuat model klasifikasi pada DNA *sequences Drosophila melanogaster* ?
2. Berapa hasil evaluasi kinerja yang didapatkan dari metode *Gated Recurrent Unit* (GRU) dalam mengklasifikasikan DNA *sequences* pada *Drosophila melanogaster* ?
3. Apakah hasil yang didapatkan lebih baik dari penelitian terdahulu oleh Beder, et al (2021)?

1.3. Batasan Masalah

Batasan masalah pada penelitian ini adalah :

1. Proses klasifikasi gen esensial didasarkan pada DNA *sequence Drosophila melanogaster* dan arsitektur model *Gated Recurrent Unit* (GRU) didasarkan pada arsitektur pada penelitian Vazhayil, R & KP (2018).
2. Klasifikasi dilakukan hanya dengan 2 kelas yaitu *essential* dan *non-essential*.
3. Data yang digunakan berupa data *Cellular Essential Gene* (CEG) dan *Organismal Essential Gene* (OEG) dari *Drosophila melanogaster* yang diperoleh dari penelitian Beder, et al (2021) yang dapat diakses melalui link https://doi.org/10.1093/nargab/lqab110.

1.4. Tujuan Penelitian

Adapun tujuan dari dilakukannya penelitian ini adalah :

1. Mengevaluasi kinerja metode *Gated Recurrent Unit* (GRU) dalam mengklasifikasi DNA *sequence.*
2. Membandingkan hasil yang diperoleh dengan penelitian sebelumnya yang menggunakan dataset yang sama.

1.5. Manfaat Penelitian

Manfaat dari penelitian ini adalah :

1. Dapat menambah wawasan mengenai metode *Gated Recurrent Unit* (GRU) dalam mengklasifikasi gen esensial berdasarkan DNA *sequence* pada *Drosophila melanogaster*.
2. Dapat mengetahui performa dan tingkat keberhasilan metode *Gated Recurrent Unit* (GRU) dalam mengklasifikasi DNA *sequence*.

II. TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Penelitian terdahulu digunakan sebagai acuan dalam melakukan perbandingan hasil klasifikasi. Penelitian pembanding memiliki topik mengenai gen esensial. Tabel 1 menunjukkan gambaran umum dari penelitian terdahulu yang menjadi acuan dalam penelitian ini.

Tabel 1. Penelitian terdahulu yang terkait dengan penelitian ini

No	Penelitian	Data	Metode	Hasil
1.	*Essential gene prediction in Drosophila melanogaster using machine learning approaches based on sequence and functional features*. (Aromolaran, et al., 2020)	OGEE & DEG *Essential gene* : 441 gen *Non Essential* gene :11.788	*Generalised Linear Model* (GLM), *Support-Vector Machines* (SVM), *Random Forest* (RF), *Artificial Neural Networks* (NNET) dan *Extreme Gradient Boosting* (XGB)	GLM -ROC-AUC = 89 % -PR-AUC = 27% -F1-*score* = 28% SVM -ROC-AUC = 88% -PR-AUC = 27% -F1-*score* = 30% NNET -ROC-AUC = 85% -PR-AUC = 20% -F1-*score* = 24% RF -ROC-AUC = 90% -PR-AUC = 29% -F1-*score* = 32% XGB -ROC-AUC = 90% -PR-AUC = 30% -F1-*score* = 34%

2.	*Identifying essential genes across eukaryotes by machine learning* (Beder, et al., 2021)	OGEE *Drosophila melanogaster* CEG :11547 *Essential gene* : 1227 *Non Essential* gene : 10320 OEG : 517 *Essential gene* : 246 *Non Essential* gene : 271	*Random Forest* (RF), *Extreme Gradient Boosting* (XGB), *Oversampling* SMOTE	CEG RF : -ROC-AUC=84% -PR-AUC = 41% -*Sensitivity* = 54% -*Specificity* = 88% XGB: -ROC-AUC=83% -PR-AUC = 40% -*Sensitivity* = 55% -*Specificity* = 86% OEG RF : -ROC-AUC=92% -PR-AUC = 92% -*Sensitivity* = 82% -*Specificity* = 82% XGB: -ROC-AUC=91% -PR-AUC =90% -*Sensitivity* = 81% -*Specificity* = 85%
3.	*Performance evaluation of features for gene essentiality prediction* (Aromolaran, et al, 2021)	OGEE & DEG *Saccharomyces. Cerevisiae* Gen Essential : 1037 *Non Essential* : 4543 *Schizosaccharomyces pombe.* *Gen Essential* : 1346 *Non Essential* : 3689	*Random Forest* (RF), *Artificial Neural Networks* (NNET), *Extreme Gradient Boosting* (XGB)	DNA *sequence S. cerevisiae* RF AUROC = 67% AUPRC = 32,3% *Precision* =54,5% *Sensitivity* = 1,7% NNET AUROC = 60,7% AUPRC = 26,1% *Precision* =31,1% *Sensitivity*=21,1% XGB AUROC = 66,3% AUPRC = 31,7% *Precision* =47,3% *Sensitivity* = 5,9% DNA *sequence S. pombe*. RF AUROC = 60,8% AUPRC = 35,1% *Precision* = 41% *Sensitivity* = 5,1%

	NNET AUROC = 54,9% AUPRC = 31,4% *Precision*=32,1% *Sensitivity*=30,2% XGB AUROC = 60,6% AUPRC = 35,8% *Precision*=38,1% *Sensitivity*=16,9%

Penjabaran mengenai masing-masing penelitian pada Tabel 1 dapat dijelaskan sebagai berikut :

2.1.1. *Essential gene prediction in Drosophila melanogaster using machine learning approaches based on sequence and functional features*

Penelitian ini dilakukan oleh Aromolaran, et al. (2020). Pada penelitian ini menggunakan gabungan dari dataset *Online Gene Essentiality database* (OGEE) dan *Database of Essential Genes* (OGE) dengan total gen esensial sebanyak 441 dan 11.788 gen non esensial untuk *Drosophila melanogaster*. Fitur yang diterapkan berasal dari 8 kategori yaitu *protein sequence*, *gene sequence*, *domains*, *topology (transcription profiles)*, *Topology (PPI)*, *evolution*, *localization*, dan *gene sets.*

Metode yang digunakan dalam penelitian ini adalah *Generalised Linear Model* (GLM), *Support-Vector Machines* (SVM), *Random Forests* (RF), *Artificial Neural Networks* (NNET) dan *Extreme Gradient Boosting* (XGB). *Metode Extreme Gradient Boosting* (XGB) memiliki akurasi yang lebih tinggi diantara metode yang digunakan dalam penelitian ini dengan nilai ROC-AUC sebesar 90%, PR-AUC sebesar 30% dan nilai F1-*score* sebesar 34%.

2.1.2. *Identifying essential genes across eukaryotes by machine learning*

Penelitian ini dilakukan oleh Beder, et al. (2021) yang bertujuan untuk mengidentifikasi gen esensial pada eukariota. Dataset diambil dari 6 organisme eukariotik diantaranya *Caenorhabditis elegans, D.melanogaster, H. sapiens, M. musculus, S. cerevisiae* dan *S. pombe* yang didapat dari OGEE dan DEG. Total data gen esensial yang didapat sebanyak 11.038 dan 67.035 untuk non-esensial dengan 7 fitur kategori yaitu *protein* dan *gene sequence, functional domains, topological features, evolution/conservation, subcellular localization*, dan *gene sets* dari *Gene Ontology.*

Pada penelitian ini, menggunakan 2 dataset yaitu CEG (*Celuller Essential Gene*) dan OEG (*Organismal Essential Gene*). Namun ada organisme yang hanya menggunakan CEG atau OEG saja maupun keduanya. Pada *D.melanogaster* menggunakan 2 dataset yaitu CEG dan OEG. Klasifikasi gen esensial dilakukan dengan metode *machine learning* yaitu *Random Forest* (RF) dan *Extreme Gradient Boosting* (XGB), dan pada penelitian ini juga dilakukan proses resampling yaitu dengan teknik SMOTE. Pada CEG, *Random Forest* menghasilkan ROC-AUC sebesar 84%, nilai PR-AUC sebesar 40%, *sensitivity* sebesar 54% dan *specificity* sebesar 88%. Sedangkan XGB memperoleh nilai ROC-AUC sebesar 83%, PR-AUC sebesar 40%, *sensitivity* sebesar 55% dan *specificity* sebesar 86%. Pada OEG, *Random Forest* menghasilkan ROC-AUC sebesar 92%, nilai PR-AUC sebesar 92%, *sensitivity* sebesar 82% dan *specificity* sebesar 82%, sedangkan XGB memperoleh nilai ROC-AUC sebesar 91%, PR-AUC sebesar 90%, *sensitivity* sebesar 81% dan *specificity* sebesar 85%.

2.1.3. *Performance evaluation of features for gene essentiality prediction*

Penelitian ini dilakukan oleh Aromolaran, et al. (2021) untuk mengevaluasi kinerja fitur untuk prediksi gen esensial. Pada penelitian

ini menggunakan 2 organisme yaitu *S.cerevisiae* dan *S.pombe* yang diperoleh dari database OGEE dan DEG. Pada *S.cerevisiae* diperoleh data gen esensial sebanyak 1037 dan non esential sebanyak 4543. Data untuk *S. pombe* diperoleh gen esential sebanyak 1346 dan non esential sebanyak 3689. Metode yang digunakan pada penelitian ini adalah *Random Forest, Artificial Neural Networks* dan *Extreme Gradient Boosting*. Kategori yang digunakan untuk melakukan prediksi adalah DNA *sequence, protein sequence, ontology* dan *topology*.

2.2. Gen Esensial

Gen adalah unit genetik dasar yang menyandikan sifat-sifat biologis (Guo, Ju, Chen, & Wang, 2021). Gen menyimpan informasi yang berkaitan dengan faktor dan proses biologis seperti ras, golongan darah, kehamilan, pertumbuhan dan proses biologis lainnya. Pada sebagian besar organisme, gen tersusun atas asam deoksiribonukleat (DNA) yang tersusun secara linear pada kromosom.

Gen yang memainkan peran yang menentukan dalam kelangsungan hidup dan perkembangan suatu organisme dalam kondisi umum disebut gen esensial (Peng, et al., 2017). Gen esensial didefinisikan sebagai gen yang diperlukan untuk kelangsungan hidup suatu organisme atau sel. Gen ini merupakan kumpulan gen minimal yang diperlukan untuk sel hidup (Zhang & Ren, 2015). Untuk bertahan hidup, suatu organisme harus mampu melakukan setidaknya dua fungsi mendasar yaitu memperoleh energi, dan bereproduksi. Apabila gen tersebut dihapus maka dapat menyebabkan kematian pada sel.

2.3. DNA *sequence*

Deoxyribonucleic acid (DNA) merupakan molekul yang membawa informasi genetik. DNA tersusun atas banyak nukleotida yang mengandung gula deoksiribosa, gugus fosfat, dan pasangan basa nitrogren yang berbentuk *double helix* terpilin. Basa nitrogen direpresentasikan dengan huruf-huruf

yaitu A (Adenin), T (Timin), G (Guanin) dan C (Sitosin). Setiap bentuk nukleotida berikatan dengan pasangan komplementernya pada untai berlawanan dalam DNA untai ganda. Adenin dan timin berpasangan, sedangkan sitosin dan guanin berpasangan (Gunasekaran, et al., 2021).

DNA *sequence* merupakan urutan basa nitrogen yang berisi informasi dasar dari suatu gen atau genom. DNA *sequence* mengandung instruksi yang dibutuhkan makhluk hidup untuk pembentukan tubuh. Dengan melakukan *sequencing* pada DNA, kode genetik dari suatu molekul dapat diketahui sehingga dapat dimanfaatkan untuk mengidentifikasi fungsi gen dengan cara membandingkan *sequence* lain yang telah diketahui (Glick, Pasternak & Patten, 2010) (Rogers, 2011).

2.4. *Drosophila melanogaster*

Drosophila melanogaster merupakan salah satu organisme yang dipelajari paling intensif dalam biologi dan berfungsi sebagai sistem model untuk penyelidikan banyak proses perkembangan dan seluler yang umum terjadi pada eukariota tingkat tinggi, termasuk manusia (Adams, et al., 2000). Menurut (Borror, et al., 1992) *Drosophila melanogaster* diklasifikasikan sebagai berikut.

Kingdom : *Animalia*
Phyllum : *Arthropoda*
Class : *Insecta*
Ordo : *Diptera*
Family : *Drosophilidae*
Genus : *Drosophila*
Species : *Drosophila melanogaster*

Drosophila melanogaster memiliki panjang tubuh berkisar antara 3-5 mm dan memiliki ciri mata berwarna merah, mata majemuk yang berbentuk bulat agak *ellips*. Pada bagian atas kepalanya terdapat mata tunggal (*oceli*) dengan

ukuran relatif lebih kecil dibandingkan mata majemuknya (Robert, 2005). Warna tubuhnya yaitu kuning kecoklatan dengan dibagian belakang bermotif cincin berwarna hitam. Sayap *Drosophila melanogaster* berasal dari thorak, vena tepi sayap (*costal vein*) yang terbagi menjadi dua bagian yang terinterupsi dekat tubuhnya. Sungutnya (*arista*) berbentuk bulu dan memiliki 7-12 percabangan. *Drosophila melanogaster* dapat dilihat pada Gambar 1.

Gambar 1. *Drosophila melanogaster* (Perveen, 2018).

Sekitar 75% dari semua gen penyakit manusia yang diketahui memiliki homolog dengan lalat termasuk yang bertanggung jawab atas gangguan perkembangan dan neurologis, kanker, penyakit kardiovaskular, penyakit metabolik dan penyimpanan, serta gen yang dibutuhkan untuk fungsi sistem visual, pendengaran dan kekebalan tubuh (Ethan, 2005).

2.5. *Tokenization*

Tokenization merupakan mekanisme pemisahan atau pemecahan kalimat dan kata menjadi morfem terkecil yang biasa disebut token (Rai & Borah, 2020). *Tokenization* memiliki tujuan yaitu memecah data yang tidak terstruktur menjadi potongan-potongan informasi numerik yang dapat digunakan untuk proses pembelajaran mesin. Token dapat berupa kata, karakter atau subkata. *Tokenization* dibedakan menjadi 3 level, yaitu sebagai berikut.

1. *Word-level tokenization*, adalah tokenisasi yang membagi teks sesuai dengan spasi antar kata. Dalam hal ini, setiap kata dalam urutan dianggap sebagai satu token.

2. *Character-level tokenization* membagi kata menjadi bagian terkecil, seperti mengubah huruf A-Z menjadi nilai 1-26. Setiap karakter dalam urutan dianggap sebagai satu token.
3. *Sub-word tokenization*, berada di antara *tokenization* tingkat kata dan tingkat karakter. Ini melibatkan pemecahan kata-kata individual menjadi bagian-bagian yang lebih kecil dan kemudian mengubah bagian-bagian yang lebih kecil tersebut menjadi angka. Dalam hal ini, setiap kata dapat dianggap sebagai beberapa token.

Pada penelitian ini menggunakan tokenisasi *character-level tokenization* untuk memproses data DNA, contoh implementasi menggunakan *character-level tokenization* ditunjukkan pada Tabel 2.

Tabel 2. Implementasi *Character-level Tokenization*

Sebelum *Tokenization*	**Sesudah *Tokenization***
ATCG	{‘A’ = 1, ‘T’=2, ‘C’=3, ‘G’=4}

Dari tokenisasi di atas diberikan data *sequence*, hasilnya dapat dilihat pada Tabel 3.

Tabel 3. Hasil *Tokenization*

Sequence	**Hasil *Tokenization***
ATGCTTTACGAT	[1 2 4 3 2 2 2 1 3 4 1 2]
CATGATCCAATC	[3 1 2 4 1 2 3 3 1 1 2 3]
ACTGATGCCTAA	[1 3 2 4 1 2 4 3 3 2 1 1]

2.6. *Padding*

Padding adalah proses yang digunakan untuk membuat setiap *input* memiliki ukuran yang sama. Data *sequence* DNA memiliki *input* yang berbeda-beda sedangkan semua vektor *input* harus memiliki ukuran yang sama untuk dimasukkan ke model. Hal ini berarti menetapkan panjang yang sama(*max_length*) untuk semua DNA dan kemudian memotong DNA yang lebih panjang sepanjang *sequence* yang ditentukan atau mengisi DNA yang lebih pendek dengan karakter "buatan" sepanjang *sequence* yang ditentukan

tersebut (Rio, et al., 2020). Karakter buatan yang biasa digunakan adalah 0. *Padding* yang diisi dengan 0 diawal *sequence* disebut *pre-padding* sedangkan *padding* yang diisi dengan 0 diakhir *sequence* disebut *post-padding*. Contoh penggunaan *padding* ditunjukkan pada Tabel 4.

Tabel 4. Implementasi *Padding*

Sequence	***pre-padding***	***post-padding***
[1,2,3,4]	[1,2,3,4]	[1,2,3,4]
[2,3]	[0,0,2,3]	[2,3,0,0]

Pada Tabel 4, *pre-padding* akan memberikan tambahan 0 diawal *sequence* sedangkan *post-padding* akan memberikan tambahan 0 diakhir *sequence*.

2.7. *Random Undersampling*

Undersampling mengacu pada teknik yang dirancang untuk menyeimbangkan distribusi kelas untuk data yang tidak seimbang. *Undersampling* merupakan teknik yang digunakan untuk menghilangkan *instance* kelas mayoritas (Hasanin & Khoshgoftaar, 2018). *Undersampling* adalah metode yang efisien untuk pembelajaran ketidakseimbangan kelas. Metode ini menggunakan sebagian dari mayoritas kelas untuk melatih model, set pelatihan menjadi lebih seimbang dan proses pelatihan menjadi lebih cepat (Liu, Wu, & Zhou, 2009). Teknik *undersampling* yang paling sederhana melibatkan pemilihan data secara acak dari kelas mayoritas dan menghapusnya dari kumpulan data pelatihan. Ini disebut sebagai *random undersampling*. *Random undersampling* didasarkan pada penghilangan kelas mayoritas secara acak (Hasanin & Khoshgoftaar, 2018).

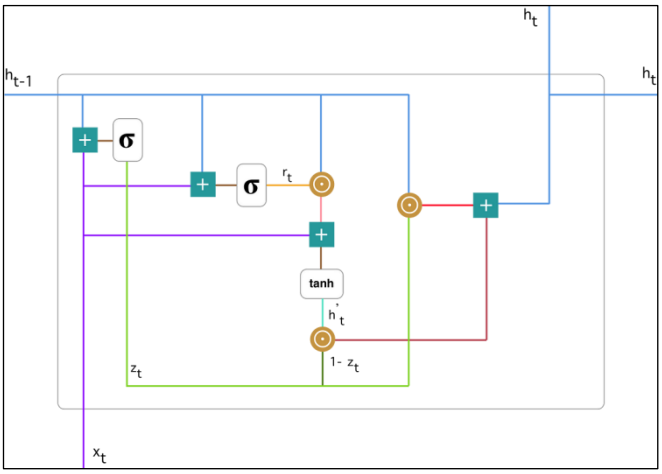
2.8. *Embedding Layer*

Embedding layer merupakan *layer* pertama yang mempelajari representasi vektor. *Embedding layer* digunakan untuk mengubah *input* diskrit ke titik-titik dalam ruang vektor, yang disebut *embedding vector*. *Embedding layer* adalah pokok dari *natural language processing* (NLP) dimana digunakan

untuk mewakili kata-kata dalam ruang berdimensi L, di mana L adalah panjang vektor (Shadab, et al., 2020). Dalam *embedding layer* terdapat 3 parameter yaitu *input* dimensi, *output* dimensi dan *input length*. Input dimensi adalah ukuran kosa kata. *Output* dimensi adalah panjang vektor dalam setiap karakter dan *input length* adalah panjang maksimum urutan.

2.9. GRU

Gated recurrent Unit (GRU) adalah metode pengembangan yang berasal dari *Recurrent Neural Network* (RNN) yang diusulkan oleh (Cho K, et. al, 2014) yang bertujuan untuk menghasilkan setiap *recurrent* unit secara adaptif menangkap hubungan (dependensi) dari skala waktu yang berbeda-beda (Chung, et al, 2014). GRU merupakan struktur berulang yang dirancang dengan hati-hati yang membentuk *trade off* yang baik antara kinerja dengan kecepatan. GRU memiliki mekanisme internal yang disebut "*gates*" yang mengatur aliran informasi. *Gates* ini dapat mempelajari data mana yang secara berurutan penting untuk disimpan atau dibuang. Dengan itu hal itu, informasi relevan yang relevan dapat diteruskan untuk membuat prediksi. GRU memiliki dua *gate* yaitu *reset* dan *update.* Gerbang *reset* (*reset gate*) merupakan *gate* yang digunakan untuk memutuskan berapa banyak informasi masa lalu yang harus dilupakan. *Update gate* merupakan gate yang memutuskan informasi apa yang akan dibuang dan informasi baru apa yang akan ditambahkan. *Gates* pada GRU ditunjukkan pada Gambar 2.



Gambar 2. Cara Kerja Arsitektur GRU (Kostadinov, 2017).

Untuk membangun model GRU langkah pertama adalah dengan menghitung *update gate* z_t menggunakan Persamaan (1).

$$z_t = \sigma\left(w^{(z)}x_t + u^{(z)}h_{t-1} + b\right) \quad (1)$$

z_t = *update gate*
σ = *activation function sigmoid*
w, u = *weight* (bobot)
x_t = *input*
h_{t-1} = *hidden state*
b = bias

Untuk persamaan *activation function sigmoid* ditunjukkan pada Persamaan (2).

$$\sigma = \frac{1}{1+ exp^{-input}} \quad (2)$$

Update gate membantu model untuk menentukan berapa banyak informasi masa lalu (dari langkah waktu sebelumnya) yang perlu diteruskan ke masa depan.

Langkah selanjutnya adalah menghitung *reset gate yang* digunakan untuk menentukan banyaknya informasi sebelumnya yang dihapus dan bagaimana menggabungkan *input* baru dan informasi sebelumnya. *Reset gate* dihitung menggunakan Persamaan (3).

$$r_t = \sigma\left(w^{(r)}x_t + u^{(r)}h_{t-1} + b\right) \quad (3)$$

r_t = *reset gate*
σ = *activation function sigmoid*
w, u = *weight* (bobot)
x_t = *input*

h_{t-1} = *hidden state*

b = bias

Kemudian menghitung kandidat *hidden state* (h'_t) yang akan menggunakan *reset gate* untuk menyimpan informasi masa lalu yang relevan. Kandidat *hidden state* dapat dihitung dengan Persamaan (4).

$$h'_t = tanh(w^{(h)}x_t + r_t \odot u^{(h)}h_{t-1} + b) \quad \text{.....(4)}$$

Dimana ⊙ *hadamard product operation* dan *tanh* adalah fungsi aktivasi, *tanh* (menekan nilai agar selalu antara -1 dan 1). Persamaan untuk *tanh* dapat dilihat pada Persamaan (5).

$$tanh = \frac{exp^{input} - exp^{-input}}{exp^{input} + exp^{-input}} \quad \text{.....(5)}$$

Operasi perhitungan r_t dengan uh_{t-1} akan menentukan apa yang harus dihapus dari langkah waktu sebelumnya. Langkah terakhir, *network* perlu menghitung h_t(*hidden state*), vektor yang menyimpan informasi saat ini dan meneruskannya ke jaringan. Untuk melakukan itu *update gate* diperlukan, hal ini menentukan apa yang dikumpulkan dari konten memori saat ini, h'_t dan dari langkah sebelumnya h_{t-1}. *Hidden state* juga merupakan *output* (y_t). Perhitungan h_t dapat dilihat pada Persamaan (6).

$$h_t = z_t \odot h_{(t-1)} + (1 - z_t) \odot h'_t \quad \text{.....(6)}$$

Metode GRU mempunyai lebih sedikit parameter sehingga komputasinya lebih sederhana dan waktu yang diperlukan juga lebih sedikit.

Contoh implementasi GRU untuk memprediksi kata berikutnya pada kata “Ma” adalah sebagai berikut.

Input teks : “MathMathMathMathMath”

Pada langkah pertama *input* diubah menjadi numerik.
'h': 0, 'a': 1, 't': 2, 'M': 3

Sehingga *input* datanya menjadi :
MathMathMathMathMath = [3,1,2,0,3,1,2,0,3,1,2,0,3,1,2,0]

Kemudian membagi panjang data menjadi 3 *sequence*.

[3,1,2, 0,3,1, 2,0,3, 1,2,0, 3,1,2, 0,3,1, 2,0]

Selanjutnya membagi data menjadi *batch size* = 2, sebagai berikut :

[3,1,2, 0,3,1, 2,0,3, 1,2,0, 3,1,2, 0,3,1, 2,0]

Karena urutan terakhir 2 0, tidak cukup untuk 1 *sequence* dan tidak cukup untuk 1 *batch*, maka tidak akan digunakan.

Dari *batch size* kemudian dibagi menjadi *mini batches* dan di *transpose.*

$$\left(\left(\begin{array}{c}[[3,1,2,\\ 0,3,1]]\end{array}\right)\left(\begin{array}{c}[[2,0,3,\\ 1,2,0]]\end{array}\right)\left(\begin{array}{c}[[3,1,2,\\ 0,3,1]]\end{array}\right)\right)$$

Hasil *transpose*

$$\left(\begin{array}{ccc}\left(\begin{bmatrix}3\\0\end{bmatrix}\right) & \begin{bmatrix}1\\3\end{bmatrix} & \left(\begin{bmatrix}2\\1\end{bmatrix}\right)\\ \left(\begin{bmatrix}2\\1\end{bmatrix}\right) & \begin{bmatrix}0\\2\end{bmatrix} & \left(\begin{bmatrix}3\\0\end{bmatrix}\right)\\ \left(\begin{bmatrix}3\\0\end{bmatrix}\right) & \begin{bmatrix}1\\3\end{bmatrix} & \left(\begin{bmatrix}2\\1\end{bmatrix}\right)\end{array}\right)$$

Langkah selanjutnya adalah *one-hot encoding*

$$\left(\begin{bmatrix}3\\0\end{bmatrix}\ \begin{bmatrix}1\\3\end{bmatrix}\ \begin{bmatrix}2\\1\end{bmatrix}\right)\left(\begin{bmatrix}2\\1\end{bmatrix}\ \begin{bmatrix}0\\2\end{bmatrix}\ \begin{bmatrix}3\\0\end{bmatrix}\right)\left(\begin{bmatrix}3\\0\end{bmatrix}\ \begin{bmatrix}1\\3\end{bmatrix}\ \begin{bmatrix}2\\1\end{bmatrix}\right) \rightarrow \left(\begin{pmatrix}0&0&0&1\\1&0&0&0\end{pmatrix}\ \begin{pmatrix}0&1&0&0\\0&0&0&1\end{pmatrix}\ \begin{pmatrix}0&0&1&0\\0&1&0&0\end{pmatrix}\right)\left(\begin{pmatrix}0&0&1&0\\0&1&0&0\end{pmatrix}\ \begin{pmatrix}1&0&0&0\\0&0&1&0\end{pmatrix}\ \begin{pmatrix}0&0&0&1\\1&0&0&0\end{pmatrix}\right)\left(\begin{pmatrix}0&0&0&1\\1&0&0&0\end{pmatrix}\ \begin{pmatrix}0&1&0&0\\0&0&1&0\end{pmatrix}\ \begin{pmatrix}0&0&1&0\\0&1&0&0\end{pmatrix}\right)$$

→ *Batch* 1

→ *Batch* 2

→ *Batch* 3

$x_{(t-1)}$ $x_{(t)}$ $x_{(t+1)}$

Batch 1

$$\left(\begin{pmatrix}0&0&0&1\\1&0&0&0\end{pmatrix}\ \begin{pmatrix}0&1&0&0\\0&0&0&1\end{pmatrix}\ \begin{pmatrix}0&0&1&0\\0&1&0&0\end{pmatrix}\right)$$

Untuk *weight matrix* diinisialisasi dengan *random* sehingga didapatkan *weight* sebagai berikut.

Hidden size = 2

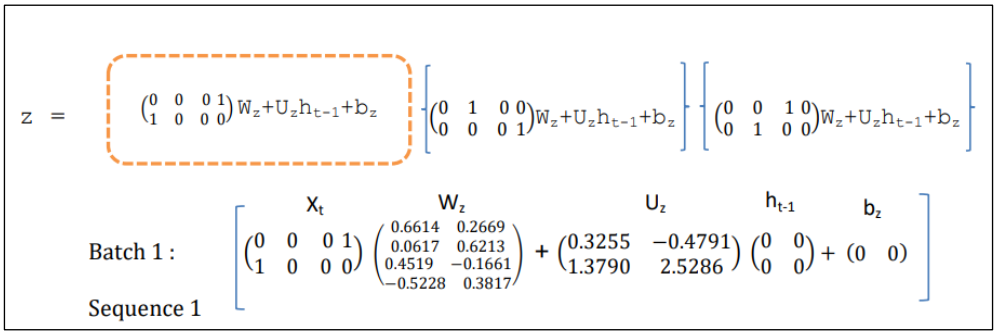
$$\left\{\begin{pmatrix}0.6614&0.2669\\0.0617&0.6213\\0.4519&-0.1661\\-1.5228&0.3817\end{pmatrix}\right\}\ \left\{\begin{pmatrix}0.3255&-0.4791\\1.3790&2.5286\end{pmatrix}\right\}$$

W_z U_z

Langkah pertama untuk membangun model GRU adalah dengan menghitung *update gate*, persamaan untuk *update gate* ditunjukkan pada Persamaan 1.

$$z = \left(\begin{pmatrix}0&0&0&1\\1&0&0&0\end{pmatrix}W_Z + U_Z h_{t-1} + b_z\right)\left(\begin{pmatrix}0&1&0&0\\0&0&0&1\end{pmatrix}W_Z + U_Z h_{t-1} + b_z\right)\left(\begin{pmatrix}0&0&1&0\\0&1&0&0\end{pmatrix}W_Z + U_Z h_{t-1} + b_z\right)$$

Visualisasi untuk *batch* 1 ditunjukkan pada Gambar 3.



Gambar 3. Visualisasi Perhitungan GRU *Batch* 1.

Update gate (z) menentukan seberapa berguna informasi masa lalu bagi keadaan saat ini. Penggunaan fungsi *sigmoid* menghasilkan nilai *update gate* antara 0 dan 1. Oleh karena itu, semakin dekat nilai *sigmoid* dengan 1, semakin banyak informasi masa lalu yang dimasukkan. Sedangkan nilai yang mendekati 0 berarti hanya informasi baru yang disimpan. Untuk perkalian *weight* dengan *input* sebagai berikut.

$$w_z x_i = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0.6614 & 0.2669 \\ 0.0617 & 0.6213 \\ 0.04519 & -0.1661 \\ -1.5228 & 0.3817 \end{pmatrix}$$

$$\rightarrow \begin{pmatrix} 0\,x\,0.6614 + 0\,x\,0.0617 + 0\,x\,0.4519 + 1\,x - 1.5228 & 0\,x\,0.2699 + 0\,x\,0.6213 + 0\,x\,0.1661 + 1\,x\,0.3817 \\ 1\,x\,0.6614 + 0\,x\,0.0617 + 0\,x\,0.4519 + 0\,x\,1.5228 & 1\,x\,0.2699 + 0\,x\,0.6213 + 0\,x\,0.1661 + 1\,x\,0.3817 \end{pmatrix}$$

$$= \begin{pmatrix} -1.5228 & 0.3817 \\ 0.6614 & 0.2699 \end{pmatrix}$$

Untuk perhitungan *hidden state* dan *weight* sebagai berikut.

$$U_z h_{(t-1)} = \begin{pmatrix} 0.3255 & -0.4791 \\ 1.3790 & 2.5286 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$\rightarrow \begin{pmatrix} 0\,x\,0.3255 + 0\,x\,-0.4791 & 0\,x\,0.3255 + 0\,x\,-0.4791 \\ 0\,x\,1.3790 + 0\,x\,2.5286 & 0\,x1.3790 + 0\,x\,2.5286 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Untuk vektor bias adalah sebagai berikut.

$$b_z = \begin{pmatrix} 0 & 0 \end{pmatrix} \rightarrow \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

Sehingga nilai z (*update gate*) yang didapatkan adalah sebagai berikut:

$$z_t = \sigma \begin{pmatrix} -1.5228 & 0.3817 \\ 0.6614 & 0.2699 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} -1.5228 & 0.3817 \\ 0.6614 & 0.2699 \end{pmatrix}$$

$$w^{(z)} x_t \qquad u^{(z)} h_{t-1} \qquad b$$

Nilai yang didapatkan diatas kemudian dipadatkan antara 0 dan 1 menggunakan *activation function sigmoid* yang ditunjukkan pada Persamaan (4). Hasil dari perhitungan sigmoid adalah sebagai berikut.

$$z_t = \sigma\begin{pmatrix} -1.5228 & 0.3817 \\ 0.6614 & 0.2699 \end{pmatrix} => \begin{pmatrix} \frac{1}{1+\ exp^{-(-1.5228)}} & \frac{1}{1+\ exp^{-(0.3817)}} \\ \frac{1}{1+\ exp^{-(0.6614)}} & \frac{1}{1+\ exp^{-(0.2699)}} \end{pmatrix}$$

$$= \begin{pmatrix} 0.1791 & 0.5943 \\ 0.6596 & 0.5663 \end{pmatrix}$$

Setelah menghitung *update gate* kemudian menghitung *reset gate*. *Reset gate* memungkinkan model untuk mengabaikan informasi masa lalu yang mungkin tidak relevan untuk waktu mendatang. Dalam setiap *batch*, *reset gate* akan mengevaluasi kembali kinerja kombinasi dari *input* sebelumnya dan *input* yang baru sesuai dengan kebutuhan *input* baru. Pada *activation function sigmoid*, nilai yang mendekati 0 berarti harus mengabaikan nilai *hidden state* sebelumnya dan sebaliknya untuk nilai yang mendekati 1. Hasil perhitungan untuk *reset gate* adalah sebagai berikut.

$$r_t = \sigma\left(w^{(r)}x_t + u^{(r)}h_{t-1} + b\right)$$

$$r_t = \begin{pmatrix} 0.6041 & 0.5664 \\ 0.2635 & 0.3628 \end{pmatrix}$$

Setelah itu menghitung kandidat *hidden state* (h'_t). Kandidat *hidden state* menggabungkan informasi dari *hidden state* sebelumnya dengan *input*.

$$w^{(h)}x_t = \begin{pmatrix} 1.5987 & -1.2770 \\ -0.4212 & -0.5107 \end{pmatrix}$$

$$r_t \odot U_h h_{t-1i} = \begin{pmatrix} 0.6041 & 0.5664 \\ 0.2635 & 0.3628 \end{pmatrix} * \begin{pmatrix} 0.4107 & -0.9880 \\ -0.9081 & 0.5423 \end{pmatrix}\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$=> \begin{pmatrix} 0.6041 & 0.5664 \\ 0.2635 & 0.3628 \end{pmatrix} * \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} => \begin{pmatrix} 0 & x & 0.6041 & 0 & x & 0.5664 \\ 0 & & 0.2635 & 0 & & 0.3628 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

$$h'_t = tanh(w^{(h)}x_t + r_t \odot u^{(h)}h_{t-1} + b)$$

$$h'_t = \begin{pmatrix} 1.5987 & -1.2770 \\ -0.4212 & -0.5107 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1.5987 & -1.2770 \\ -0.4212 & -0.5107 \end{pmatrix}$$

Nilai dalam matriks yang dihasilkan kemudian dipadatkan antara -1 dan 1 menggunakan *activation function tanh*. Perhitungan untuk *tanh* sebagai berikut.

$$h'_t = tanh\begin{pmatrix} 1.5987 & -1.2770 \\ -0.4212 & -0.5107 \end{pmatrix}$$

$$= \begin{pmatrix} \dfrac{exp^{1.5987} - exp^{-1.5987}}{exp^{1.5987} + exp^{-1.5987}} & \dfrac{exp^{-(-1.2770)} - exp^{-(-1.2770}}{exp^{-(-1.2770} + exp^{-(-1.2770} \\ \dfrac{exp^{-0.4212} - exp^{-(-0.4212)}}{exp^{-(0.4212)} + exp^{-(-0.4212)}} & \dfrac{exp^{-0.5107} - exp^{-(-0.5107)}}{exp^{-0.5107} + exp^{-(-0.5107)}} \end{pmatrix}$$

$$= \begin{pmatrix} 0.9215 & -0.8557 \\ -0.3979 & -0.4705 \end{pmatrix}$$

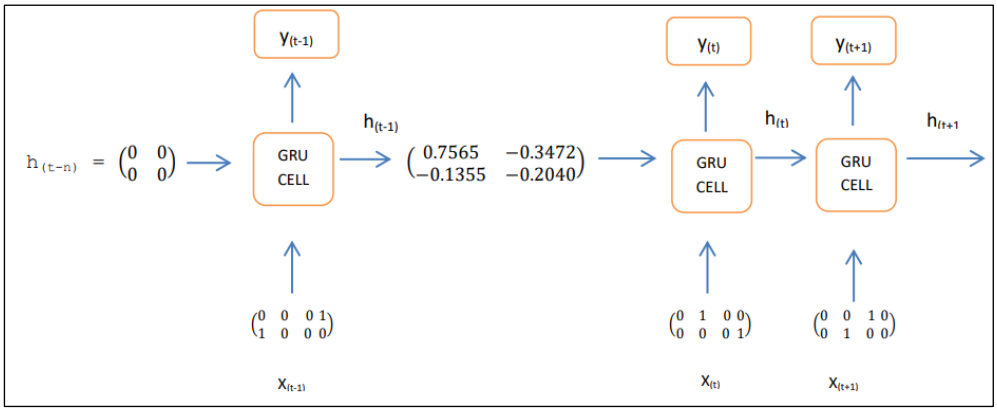
Langkah selanjutnya adalah menghitung *hidden state* yang ditunjukkan sebagai berikut.

$$h_t = z_t \odot h_{(t-1)} + (1 - z_t) \odot h'_t$$

$$h_t = \underset{z_t}{\begin{pmatrix} 0.1791 & 0.5943 \\ 0.6596 & 0.5663 \end{pmatrix}} * \underset{h_{(t-1)}}{\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}} + \underset{(1-z_t)}{\left(1 - \begin{pmatrix} 0.1791 & 0.5943 \\ 0.6596 & 0.5663 \end{pmatrix}\right)} * \underset{h'_t}{\begin{pmatrix} 0.9215 & -0.8557 \\ -0.3979 & -0.4705 \end{pmatrix}}$$

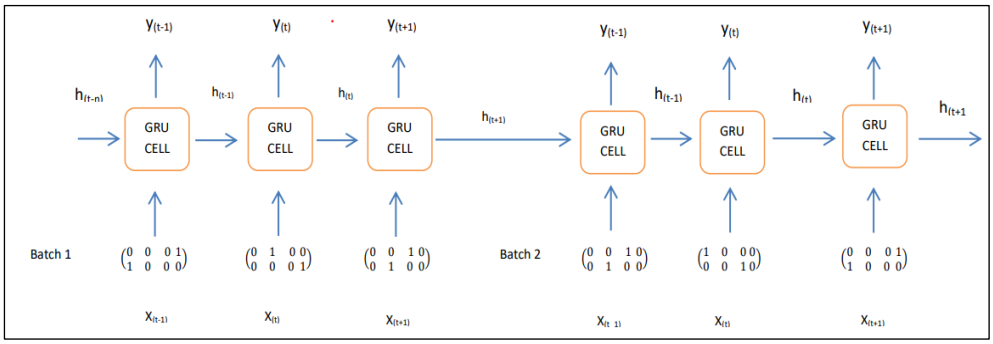
$$= \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0.7565 & -0.3472 \\ -0.1355 & -0.2040 \end{pmatrix} = \begin{pmatrix} 0.7565 & -0.3472 \\ -0.1355 & -0.2040 \end{pmatrix}$$

Setelah iterasi pertama, *hidden state* baru sekarang akan digunakan sebagai $h_{(t-1)}$ untuk *hidden state* selanjutnya. Ilustrasi tersebut ditunjukkan oleh Gambar 4.



Gambar 4. Ilustrasi GRU *Cell Hidden State.*

Perhitungan diatas akan berulang untuk *batch-batch* selanjutnya. Ilustrasi tersebut dapat dilihat pada Gambar 5.



Gambar 5. Ilustrasi GRU *Cell Batch* Selanjutnya.

Sehingga didapatkan nilai *hidden state* pada setiap *input* dari *batch* masing-masing adalah sebagai berikut.

$$\begin{pmatrix} \begin{pmatrix} 0.7565 & -0.3472 \\ -0.1355 & -0.2040 \end{pmatrix} \\ \begin{pmatrix} -0.1535 & -0.5712 \\ 0.7664 & -0.5062 \end{pmatrix} \\ \begin{pmatrix} 0.7495 & -0.8616 \\ -0.2399 & -0.6680 \end{pmatrix} \end{pmatrix} \rightarrow \textit{batch } 1$$

$$\left(\begin{pmatrix} 0.9491 & -0.9869 \\ -0.7454 & -0.0868 \end{pmatrix} \begin{pmatrix} 0.1406 & -0.9392 \\ 0.4630 & -0.4591 \end{pmatrix} \begin{pmatrix} 0.8373 & -0.9050 \\ 0.0556 & -0.6569 \end{pmatrix}\right) \rightarrow \textit{batch } 2$$

$$\left(\begin{pmatrix} 0.9054 & -0.9849 \\ -0.2446 & -0.5224 \end{pmatrix} \begin{pmatrix} -0.4335 & -0.8752 \\ 0.7853 & -0.6418 \end{pmatrix} \begin{pmatrix} 0.7948 & -0.8400 \\ -0.3061 & -0.7358 \end{pmatrix}\right) \rightarrow \textit{batch } 3$$

Selanjutnya adalah menghitung prediksi dari setiap waktu t. Untuk melakukan prediksi untuk setiap langkah waktu, langkah pertama adalah mengubah *output* menggunakan *linear layer*. *Input* yang berikan pada *input* awal berupa *input* yang terdiri dari 4 karakter unik, karena itu *output* yang diharapkan juga memiliki ukuran yang sama. Untuk itu digunakan *dense layer* atau *fully connected layer* untuk mentransformasikan *output* tersebut agar memilki ukuran yang sama dengan *input*. *Layer* tersebut kemudian diteruskan ke *activation function*, *activation function* yang digunakan yaitu *softmax*. Persamaan untuk menghitung *linear layer* ditunjukkan pada Persamaan (7).

$$Linear = W_y h_{(t-1)} + b_y \quad \text{.....(7)}$$

Perhitungan untuk memprediksi *output* dimulai dari mengubah menjadi *linear* adalah sebagai berikut.

$$Linear = \begin{pmatrix} 0.7565 & -0.3472 \\ -0.1355 & -0.2040 \end{pmatrix} \begin{pmatrix} 0.8310 & -0.2477 & -0.8029 & 0.2366 \\ 0.2857 & 0.6898 & -0.6331 & 0.8795 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}$$
$$= \begin{pmatrix} 0.5295 & -0.4269 & -0.3876 & -0.1264 \\ -0.1709 & -0.1072 & 0.2379 & -0.2115 \end{pmatrix}$$

Hasil *linear* untuk setiap *batch* adalah sebagai berikut.

$$\left(\begin{pmatrix} 0.5295 & -0.4269 & -0.3876 & -0.1264 \\ -0.1709 & -0.1072 & 0.2379 & -0.2115 \end{pmatrix} \begin{pmatrix} -0.2908 & -0.3560 & 0.4849 & -0.5387 \\ 0.4922 & -0.5390 & -0.2949 & -0.2639 \end{pmatrix} \begin{pmatrix} 0.3767 & -0.7800 & -0.0563 & -0.5805 \\ -0.3902 & -0.4014 & 0.6155 & -0.6443 \end{pmatrix}\right) \rightarrow \boxed{\textit{Batch } 1}$$

$$\left(\begin{pmatrix} 0.5068 & -0.9159 & -0.1373 & -0.6434 \\ -0.6442 & 0.1248 & 0.6534 & -0.2527 \end{pmatrix} \begin{pmatrix} -0.1515 & -0.6827 & 0.4817 & -0.7928 \\ 0.2536 & -0.4314 & -0.0811 & -0.2942 \end{pmatrix} \begin{pmatrix} 0.4373 & -0.8317 & -0.0994 & -0.5978 \\ -0.1415 & -0.4669 & 0.3713 & -0.5646 \end{pmatrix}\right) \rightarrow \boxed{\textit{Batch } 2}$$

$$\left(\begin{pmatrix} 0.4710 & -0.9037 & -0.1035 & -0.6520 \\ -0.3525 & -0.2998 & 0.5271 & -0.5173 \end{pmatrix} \begin{pmatrix} -0.6103 & -0.4963 & 0.9021 & -0.8723 \\ 0.4692 & -0.6372 & -0.2242 & -0.3786 \end{pmatrix} \begin{pmatrix} 0.4205 & -0.7763 & -0.1064 & -0.5507 \\ -0.4646 & -0.4318 & 0.7116 & -0.7196 \end{pmatrix}\right) \rightarrow \boxed{\textit{Batch } 3}$$

Terakhir adalah menetapkan aktivasi *softmax* untuk menormalkan *output* menjadi distribusi probabilitas yang berjumlah 1. Fungsi *softmax* ditunjukkan pada Persamaan (8).

$$softmax(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad \text{.....(8)}$$

Pada *softmax* dapat digunakan trik *softmax max* yang mengurangi nilai maksimum dari seluruh kumpulan data untuk mencegah nilai meledak (*exploding value*) untuk jumlah *y_lineary* atau *fully_connected* yang besar. Nilai maksimum yang didapatkan adalah 0,9021, yang akan dikurangi terlebih dahulu sebelum menerapkannya pada persamaan *softmax*.

Perhitungan untuk fungsi *softmax* adalah sebagai berikut.

1. Kurangi nilai dari seluruh elemen dengan nilai maksimum

$$\exp(y_{linear} - y_{linear\_max}) = exp\begin{pmatrix} 0.5295 & -0.4269 & -0.3876 & -0.1264 \\ -0.1709 & -0.1072 & 0.2379 & -0.2115 \end{pmatrix} - 0.9021$$

$$= \exp\begin{pmatrix} 0.5295 - 0.9021 & -0.4269 - 0.9021 & -0.3876 - 0.9021 & -0.1264 - 0.9021 \\ -0.1709 - 0.9021 & -0.1072 - 0.9021 & 0.2379 - 0.9021 & -0.2115 - 0.9021 \end{pmatrix}$$

$$= \begin{pmatrix} exp^{-0.3727} & exp^{-1.3290} & exp^{-1.2898} & exp^{-1.0285} \\ exp^{-1.0730} & exp^{-1.0093} & exp^{-0.6642} & exp^{-1.1136} \end{pmatrix}$$

$$= \begin{pmatrix} 0.6889 & 0.2648 & 0.2753 & 0.3575 \\ 0.3420 & 0.3645 & 0.5147 & 0.3284 \end{pmatrix}$$

2. Jumlahkan seluruh elemen matriks

$$\sum exp^{\exp(y_{linear} - y_{linear\_max})} = \begin{pmatrix} 0.6889 & + & 0.2648 & + & 0.2753 & + & 0.3575 \\ 0.3420 & + & 0.3645 & + & 0.5147 & + & 0.3284 \end{pmatrix}$$

$$= \begin{pmatrix} 1.5865 \\ 1.5495 \end{pmatrix}$$

3. Bagi setiap elemen dalam matriks dari langkah 1 dengan nilai dari langkah 2.

$$Softmax = \begin{pmatrix} \frac{0.6889}{1.5865} & \frac{0.2648}{1.5865} & \frac{0.2753}{1.5865} & \frac{0.3575}{1.5865} \\ \frac{0.3420}{1.5495} & \frac{0.3645}{1.5495} & \frac{0.5147}{1.5495} & \frac{0.3284}{1.5495} \end{pmatrix} = \begin{pmatrix} 0.4342 & 0.1669 & 0.1735 & 0.2254 \\ 0.2207 & 0.2352 & 0.3322 & 0.2119 \end{pmatrix}$$

$$\begin{matrix} h & a & t & M \end{matrix}$$
$$\begin{pmatrix} 0.4342 & 0.1669 & 0.1735 & 0.2254 \\ 0.2207 & 0.2352 & 0.3322 & 0.2119 \end{pmatrix}$$

Dari hasil tersebut, didapatkan bahwa probalititas tertinggi adalah huruf h, sehingga hasil prediksi setelah ‘Ma’ adalah huruf h.

2.10. *Flatten*

Flatten merupakan lapisan yang digunakan untuk membuat masukan multidimensi menjadi satu dimensi. *Flatten* akan mengubah matriks konteks yang diperoleh dari lapisan sebelumnya menjadi vektor konteks, dan kemudian akan dihubungkan ke lapisan akhir (Ahmad, Asghar, Alotaibi, & Khan, 2020).

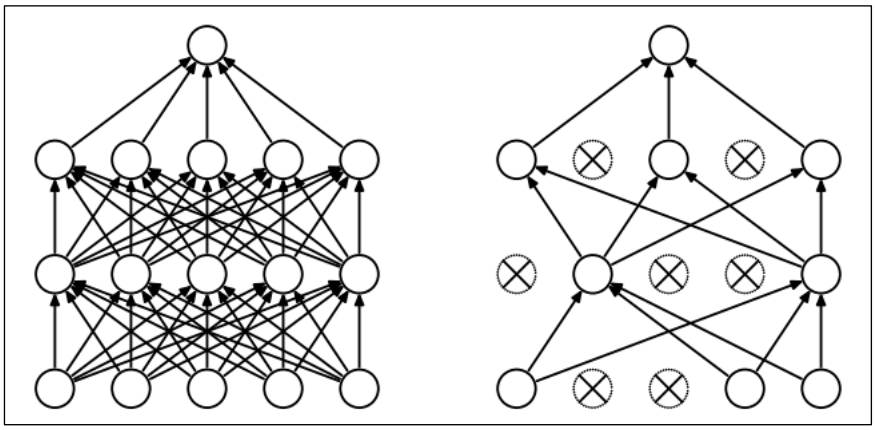
2.11. *Dense Layer*

Dense layer atau biasa disebut dengan *fully connected layer* merupakan *layer* yang digunakan pada tahap akhir *neural network*. *Layer* ini membantu

mengubah dimensi *output* dari *layer* sebelumnya sehingga model dapat dengan mudah menentukan hubungan antara nilai data tempat model bekerja. *Dense layer* pada tahap terakhir digunakan untuk mengklasifikasikan fitur *input* ke dalam kelas (Ullah, et al., 2022).

2.12. *Dropout*

Dropout adalah metode yang digunakan untuk mencegah terjadinya *overfitting* dengan menghilangkan beberapa *neuron* dari jaringan secara acak. Menurut Srivastava, et al., (2014) istilah *dropout* mengacu pada menghapus unit (*hidden* dan *visible*) dalam *neural network*, dengan mengeluarkan unit atau menghapus sementara bersama dengan semua koneksi yang masuk atau keluar dari unit tersebut. Dalam proses menghapus *node* secara acak, ini membuat jaringan kurang bergantung pada satu *node* dan dengan demikian mengurangi *overfitting* seperti yang ditunjukkan pada Gambar 6.



a b

Gambar 6. *Dropout* (Srivastava, et al., 2014).

Pada gambar diatas, gambar 6a merupakan neural network yang memiliki 2 *hidden layer* sedangkan gambar 6b merupakan gambar *neural network* yang diberikan *dropout*, yang ditandai dengan *x* pada *node*.

2.13. *Confusion Matrix*

Confusion matrix merupakan salah satu metode yang digunakan untuk mengukur performa klasifikasi. Matriks tersebut menampilkan prediksi dari klasifikasi dan klasifikasi yang aktual. Pengukuran kinerja pada *confusion matrix* terdapat 4 (empat) istilah yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP) dan *False Negative* (FN). Evaluasi *confusion matrix* dapat dilihat pada Tabel 5.

Tabel 5. *Confusion matrix*

	Actual Value	
Predicted Value	TP (*True Possitive*)	FP (*False Possitive*)
	FN (*False Negative*)	TN (*True Negative*)

Adapun istilah yang digunakan pada *confusion matrix*, diantaranya.

a. *True Positive* (TP): jumlah data positif yang diklasifikasikan dengan benar oleh sistem.
b. *True Negative* (TN): jumlah data negatif yang diklasifikasikan dengan benar oleh sistem.
c. *False Negative* (FN): jumlah data positif yang diklasifikasikan dengan salah oleh sistem.
d. *False Positive* (FP): jumlah data negatif yang diklasifikasikan dengan salah oleh sistem.

2.14. ROC-AUC

Kurva ROC (*Receiver Operating Characteristic curve*) adalah grafik yang menunjukkan kinerja model klasifikasi di semua *threshold* klasifikasi. Secara grafis kurva ROC adalah representasi hubungan dari sensitifitas dan spesifitas (Erke & Pattynama, 1998). Kurva ini memplot dua parameter yaitu :

1. *True Positive Rate* (TPR)

   Persamaan untuk TPR dapat dilihat pada Persamaan (9).

$$TPR = \frac{TP}{TP+FN} \quad \text{.....(9)}$$

TP = True Positive
FN = False Negative

TPR diperoleh dari nilai *true positive* yang dibagi dengan jumlah *true positive* dan *false negative*. TPR juga disebut dengan *sensitivity/recall* yang merupakan ukuran kebaikan model yang berguna untuk mengukur seberapa baik model terbentuk untuk memprediksi secara tepat data *testing* di kelas positif yang tepat terprediksi ke dalam kelas positif (Werdhana, 2017).

2. *False Positive Rate* (FPR)
   Persamaan untuk FPR dapat dilihat pada Persamaan (10).

$$FPR = \frac{FP}{FP+TN} \quad \text{.....(10)}$$

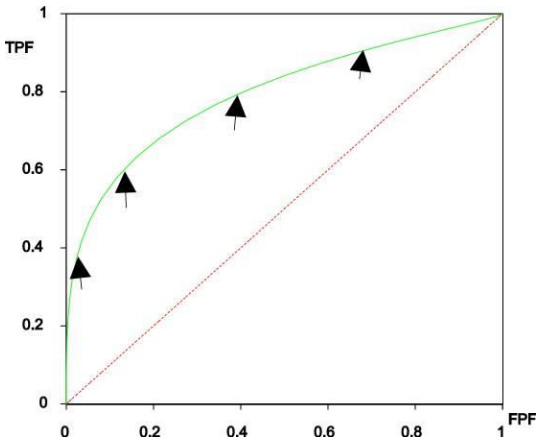
FP = False Positive
TN = True Negative

FPR diperoleh dari nilai *false positive* yang dibagi dengan jumlah *true negative* dan *false positive*. FPR yaitu proporsi kasus negatif yang salah diprediksi sebagai positif.
FPR bisa didapat dari *specificity* yaitu, 1 – *specificity*. Untuk persamaan *specificity* ditunjukkan pada Persamaan (11).

$$Specificity = \frac{TN}{FP+TN} \quad \text{.....(11)}$$

Specificity merupakan ukuran kebaikan model yang berguna untuk mengukur seberapa baik model untuk memprediksi secara tepat data *testing* di kelas negatif yang tepat terprediksi ke dalam kelas negatif.

ROC-AUC dikatakan baik jika nilai TPR semakin tinggi dan nilai FPR semakin rendah. TPR semakin tinggi berarti banyak kelas positif yang diklasifikasikan dengan benar oleh model. Nilai FPR semakin kecil berarti nilai semakin kecil kesalahan model yang memprediksi kelas negatif, dan berarti kelas negatif yang diprediksi benar oleh model semakin banyak. Jika digambarkan oleh suatu kurva, kinerja algoritma dikatakan baik apabila kurva mendekati titik (0,1) dan dikatakan buruk apabila kurva tersebut mendekati garis *baseline* atau garis yang melintang dari titik (0,0). Contoh kurva ROC ditunjukkan pada Gambar 7, di mana TPR berada pada sumbu *y* dan FPR berada pada sumbu *x*.



Gambar 7. Contoh Kurva ROC (Tilaki, 2013).

Luas area di bawah kurva mampu mengukur seberapa besar kemampuan model untuk mendiskriminasi observasi yang mengalami kejadian sukses dan tidak sukses. Luas area dibawah kurva disebut dengan AUC (*Area Under the Curve*). AUC merupakan metode yang juga digunakan untuk menghitung performa klasifikasi, dengan memanfaatkan area dibawah kurva ROC untuk menghitung performanya. Nilai AUC merupakan suatu angka yang menangkap kinerja keseluruhan model. Nilai AUC berkisar antara 0-1, dengan nilai yang lebih tinggi menunjukkan performa model yang lebih baik. AUC mampu menangani kelemahan *sensitivity* dan *specificity* yang cenderung tidak selaras apabila kelas respon *imbalance* (Werdhana, 2017).

2.15. PR-AUC

PR-AUC (*Precision Recall Area Under Curve*) adalah metrik yang digunakan untuk mengevaluasi kinerja algoritma klasifikasi biner, yang memberikan gambaran lebih informatif untuk data yang *imbalance* (tidak seimbang) (Davis & Goadrich, 2006). PR-AUC dihitung sebagai area di bawah kurva *precision-recall*, dimana setiap titik pada kurva ditentukan oleh nilai *threshold* yang berbeda untuk mengubah prediksi kontinu menjadi biner (Sofaer, Hoeting, & Jarnevich, 2018). Kurva *precision-recall* membantu memvisualisasikan bagaimana pilihan *threshold* memengaruhi kinerja pengklasifikasi, dan bahkan dapat membantu memilih *threshold* terbaik untuk masalah tertentu. Kurva *precision-recall* dibangun dengan menghitung dan memplot *precision* terhadap *recall* untuk satu *classifier*. Untuk perhitungan *precision* ditunjukkan oleh Persamaan (12) dan *recall* ditunjukkan oleh Persamaan (13).

$$Precision = \frac{TP}{TP+FP} \quad \text{.....(12)}$$

Dimana TP adalah *true positive* dan FP adalah *false positif*. *Precision* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positif.

$$Recall = \frac{TP}{TP+FN} \quad \text{.....(13)}$$

Dimana TP adalah *trus positive* dan FN adalah *false negatif*. *Recall* merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif.

Dalam mengevaluasi data tidak seimbang PR-AUC lebih baik daripada ROC-AUC. ROC memplot nilai FPR, jika nilai tersebut semakin kecil maka semakin baik pula model dalam mengklasifikasikan kelas negatif. Namun dalam data yang tidak seimbang, jika negatif yang menjadi kelas mayoritas nilai FPR akan tetap kecil karena penyebut akan tetap besar. Penyebut dalam

FPR adalah nilai FP dan TN, yang merupakan jumlah kelas negatif aktual. PR-AUC merupakan metrik yang memplot *precision-recall*, dimana *precision* dan *recall* adalah metrik yang fokus terhadap kelas positif, sehingga metrik ini tidak terpengaruh terhadap ketidakseimbangan kelas.

III. METODE PENELITIAN

3.1. Tempat dan Waktu Penelitian

Berikut adalah pemaparan tempat penelitian dan waktu serta jadwal penelitian :

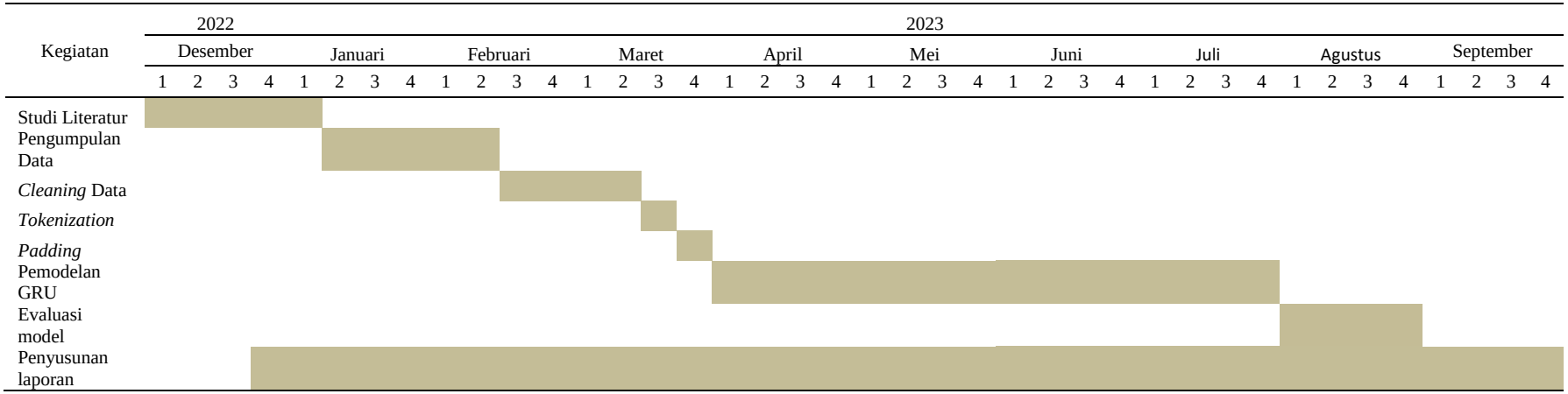
3.1.1. Tempat Penelitian

Penelitian dilakukan di Lab RPL Jurusan Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung.

3.1.2. Waktu dan Jadwal Penelitian

Penelitian dilakukan pada bulan Desember 2022 di semester tujuh ganjil hingga penyelesaian pada bulan September 2023. Alur waktu pengerjaan dapat dilihat pada Tabel 6.

Tabel 6. Alur Waktu Pengerjaan Penelitian



Pada Tabel 6 menjelaskan tentang alur waktu pengerjaan penelitian, yang secara umum dibagi menjadi 3 tahapan yaitu :

1. Tahap Awal Penelitian

   Tahap ini merupakan tahapan awal penelitian yang terdiri dari tahapan studi literatur, dan pengumpulan data. Data yang diperoleh merupakan dataset yang diambil dari penelitian (Beder, et al., 2021) berupa DNA *sequence.* Data tersebut dapat diakses pada link berikut.

   https://doi.org/10.1093/nargab/lqab110.

2. Pelaksanaan Penelitian

   Pada tahap ini merupakan tahapan untuk memulai penelitian yang dimulai dari tahap *preprocessing* data dan kemudian dilanjutkan dengan tahap pemodelan. Tahap *preprocessing* yang dilakukan berupa *cleaning* data, penggabungan data berlabel dengan *sequenc*e, *tokenization* serta *padding*. Setelah proses *preprocessing* dilanjutkan dengan tahap pemodelan menggunakan GRU.

3. Evaluasi

   Tahap evaluasi merupakan tahap terakhir dalam penelitian. Pada tahap ini mengevaluasi model yang dibuat.

3.2. Data dan Alat

3.2.1. Data

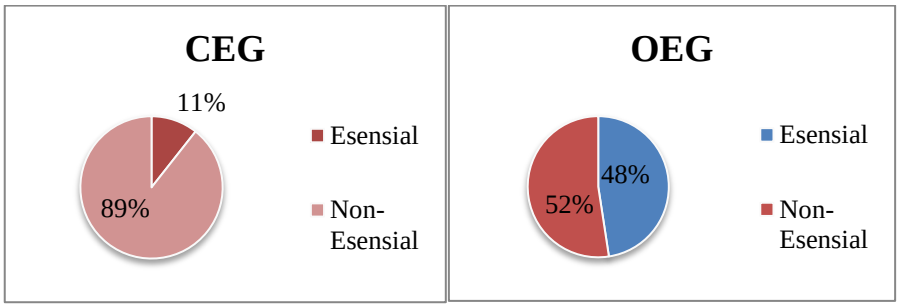
Data yang digunakan merupakan dataset DNA *sequence* dari *Drosophila melanogaster* yang diperoleh dari penelitian (Beder, et al., 2021). Dataset ini terdiri dari dataset CEG (*Celuller Essential Gene*) dan OEG (*Organismal Essential Gene*). Dataset CEG merupakan dataset gen esensial yang terlibat dalam proses biogenesis, makromolekul seluler dan siklus sel atau poliferasi sedangkan OEG adalah dataset gen esensial yang terlibat dalam proses pengayaan dalam regulasi, perkembangan atau morfogenesis, proses yang terkait saraf serta persinyalan (Beder, et al., 2021). Pada dataset ini terdapat

12.429 DNA *sequence Drosophila melanogaster*. Jumlah dan jenis data dapat dilihat pada Tabel 7.

Tabel 7. Jumlah dan Jenis Data

Jenis Data	Gen esensial	Non esensial	Jumlah
CEG	1227	10320	11547
OEG	246	271	517

Berdasarkan Tabel 7, persentase gen esensial dan non-esensial pada dataset CEG dan OEG dapat dilihat pada Gambar 8.



Gambar 8. Persentase Jumlah *Esensial* dan *Non-Esensial*.

Pada dataset CEG, gen yang diidentifikasi sebanyak 11547, yang terdiri dari 1227 gen esensial dan 10320 non-esensial. *Sequence* terpendek pada CEG adalah berjumlah 210 dan terpanjang adalah 394.149. Pada OEG gen yang diidentifikasi sebanyak 517 yang terdiri dari 246 gen esensial dan 271 non-esensial. Panjang *seqeunce* terpendek pada OEG adalah 355 dan terpanjang adalah 167.328. Berikut bentuk dataset yang dapat dilihat pada Gambar 9.

	dna	gene
0	GTTCAATCTTTGTTTTCGTAGCGCGGCGGTCGCATCGGAGTCGAGA...	Non-essential
1	ACAGACAGCGGAGAACTCGCACACCATTCAGTTCACTCGATGACGC...	Non-essential
2	TATTGCGCCATAAACGTTTCGCTGCTCGTAACGCCACAACGCTCGA...	Essential
3	TTAGTTACCTTCCGATCGGAAGAAGAACCCGGCTGACATTAGGAAT...	Non-essential
4	ACTATCGTTATCGAGACTTCGAAGCTTTGTGTGGTTATCAAACAGG...	Non-essential

Gambar 9. Dataset DNA *Sequences*.

3.2.2. Alat

3.2.2.1. Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan dalam penelitian ini adalah :

a. *Processor* : 11th Gen Intel® Core™ i7-1165G7 @2.80GHz (8 CPUs), ~2.8GHz
b. RAM : 8 GB
c. *Strorage* : SSD 512 GB
d. *Network Interface* : Intel® Wireless Wi-Fi 6 AX201
e. *Video Graphics Array* (VGA) : Intel® Iris ® Xe Graphics

3.2.2.2. Perangkat Lunak (*Software*)

a. *Operating System* : Windows 10 Home Single Language 64-bit
b. *Tools*
    a) Google Colab

    Google Colab adalah sebuah *executable* dokumen yang dapat digunakan untuk menulis, menyimpan program melalui Google Drive. *Software* ini serupa dengan Jupyter Notebook gratis berbentuk *cloud* yang dijalankan menggunakan *browser* seperti Google Chrome dan Mozilla Firefox. Google Colab dapat digunakan untuk data *cleaning* dan transformasinya, simulasi angka, visualisasi data, pemodelan statistis, *machine learning*.

    b) *Python*

    Python merupakan bahasa pemrograman yang sering digunakan pada pemrosesan data. Salah satu kelebihan *python* adalah kemudahan dalam penggunaan dan sintak yang menyerupai bahasa alami manusia. Kelebihan lainnya adalah banyaknya

dukungan *library* yang mendukung pemrograman pada *python*.

c. *Packages*

a) *Pandas*

Pandas merupakna *library* yang yang digunakan untuk menyimpan data dalam bentuk tabulasi. *Library* ini juga dapat mempermudah dalam pemrosesan data seperti pada saat pembacan file ke dalam program.

b) *Scikit-learn*

Scikit-learn merupakan modul yang dibangun berdasarkan *Numpy, SciPy* dan *Matplotlib*. *Scikit-learn* memudahkan dalam *processing data* ataupun *training*. *Library* ini juga biasa digunakan untuk membagi data (*split* data) untuk *training* dan *testing*.

c) *Numpy*

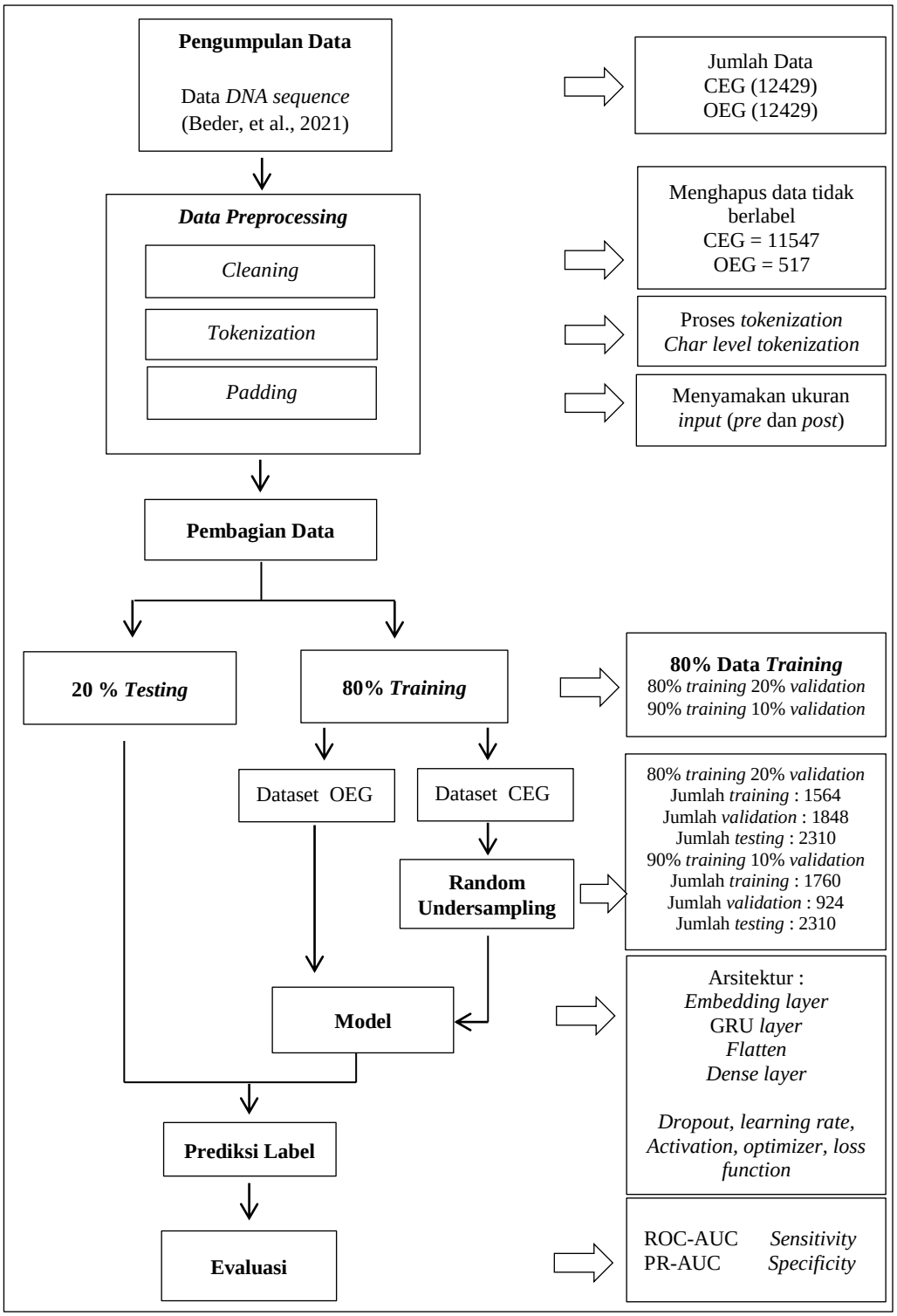
Numpy merupakan salah satu *library Python* yang berfungsi untuk proses komputasi numerik. *Numpy* mengolah data berupa *array*. *Library pandas* juga menggunakan *numpy* untuk menyimpan data *array* pada tabelnya.

d) *Tensorflow*

Tensorflow merupakan *library* yang dikembangkan oleh google. *Tensorflow* mendukung data berupa tensor, yang merupakan *array* multidimensi dengan dimensi yang lebih besar. *Array* dengan beberapa dimensi sangat berguna untuk mengelola volume data yang sangat besar.

3.3. Metodologi

Alur kerja penelitian ini berdasarkan penelitian sebelumnya. Alur kerja penelitian ini dapat dilihat pada Gambar 10.



Gambar 10. Alur kerja penelitian klasifikasi gen esensial pada *Drosophila melanogaster*.

Berdasarkan alur penelitian pada Gambar 10, berikut penjelasan setiap tahap.

1. *Data Collecting*

   Data DNA *sequence* untuk *Drosophila melanogaster* didapatkan dari penelitian (Beder, et al., 2021) terdiri dari 12429 *sequence* DNA yang dikelompokknya menjadi 2 dataset yaitu CEG dan OEG. Pada CEG, gen yang diidentifikasi sebanyak 11547, yang terdiri dari 1227 gen esensial dan 10320 non esensial. Pada OEG gen yang diidentifikasi sebanyak 517 yang terdiri dari 246 gen esensial dan 271 non-esensial.

2. *Preprocessing*

   Pada tahap ini, dilakukan *cleaning* untuk memastikan bahwa data siap untuk digunakan. Data *cleaning* dilakukan dengan penghapusan *missing value* pada dataset CEG dan OEG sehingga menyisakan data-data yang memiliki label. Selanjutnya adalah tahapan untuk representasi *sequence* yang dilakukan 2 tahap, yaitu pertama melakukan *tokenization* pada data *sequence* DNA dengan tipe *char-level tokenization* dan tahap selanjutnya adalah memberikan *padding* yang bertujuan untuk menyamakan ukuran *input* sebelum masuk ke dalam model. Pada dataset CEG memiliki proses tambahan yaitu dilakukan *undersampling* dengan teknik *random undersampling*. Hal tersebut karena pada dataset CEG memiliki data yang tidak seimbang. *Random undersampling* dilakukan setelah proses *padding*.

3. Pembagian Data

   Pembagian data merupakan tahap dimana data akan dibagi menjadi data *training*, *validation* dan *testing*. Data *training* adalah data yang digunakan untuk membangun melatih atau menghasilkan model. Data *validation* adalah data yang digunakan untuk mengoptimasi saat melatih model atau data yang digunakan untuk menguji kinerja model pada saat *training*. Data *testing* adalah data yang digunakan untuk menguji model setelah proses *training* selesai, data ini merupakan data *unseen* yang berarti data baru yang belum pernah dilihat sebelumnya. Pada penelitian ini data dibagi sebanyak 80% untuk data *training* dan 20% untuk data

testing pada masing-masing dataset. Data *training* kemudian dibagi lagi menjadi 2 skema pembagian yaitu sebanyak 80% *training* 20% *validation* dan 90% *training* 10% *validation.*

4. Klasifikasi

   Tahapan selanjutnya adalah melakukan pemodelan atau klasifikasi. Pemodelan dilakukan untuk masing-masing dataset. Pemodelan dilakukan dengan algoritma GRU, dengan beberapa *layer* yaitu, *embedding layer*, GRU *layer*, *dense layer* dan *dropout.* Parameter yang akan digunakan menggunakan rujukan dari penelitian oleh Vazhayil, et al. (2018) ditunjukkan pada Tabel 8.

Tabel 8. Parameter Model

Type	**Jumlah/ukuran**
Embedding	128
GRU	128
Dropout	0.2
Dense	2

Layer embedding merupakan lapisan pertama setelah proses *padding*. Dalam *embedding layer* terdapat 3 parameter yaitu *input* dimensi, *output* dimensi dan *input length*. *Input* dimensi adalah ukuran kosa kata dimana pada penelitian ini memiliki 4 karakter nukleotida (A, T, G, C) dan karakter *padding* (0) sehingga *input* dimensinya adalah 5. *Output* dimensi adalah panjang vektor dalam setiap karakter, pada rujukan arsitektur pada Tabel 8, jumlah *output* dimensinya adalah 128. *Input length* adalah panjang maksimum urutan, panjang maksimum urutan ini mengikuti jumlah panjang maksimum pada *padding*. Pada lapisan GRU pada Tabel 8, *neuron* yang digunakan adalah 128 yang diikuti jumlah *dropout* 0.2. Selanjutnya diikuti dengan *fully connected layer* (*dense layer*) dengan jumlah kelasnya yaitu 2.

5. Evaluasi

   Tahapan terakhir yang dilakukan adalah dengan menghitung performa dari model yang diterapkan yaitu dengan menghitung ROC-AUC, PR-AUC, *sensitivity* dan *specificity*. Dalam memilih model yang baik dalam penelitian ini memiliki indikasi yaitu :

   a. Pertama dengan melihat hasil PR-AUC, karena PR-AUC memplot nilai *precision* dan *recall*, dimana *precision* mengukur keakuratan prediksi positif dan *recall* mengukur kemampuan untuk mengidentifikasi kejadian positif dengan benar. Dengan menggunakan PR-AUC, model akan berfokus pada penilaian kelas positif dibandingkan kelas negatif.
   b. Kedua dengan mempertimbangkan nilai ROC-AUC juga dalam mengevaluasi model klasifikasi, dimana ROC memplot *True Positive Rate* (TPR) yaitu probabilitas bahwa sampel positif diprediksi dengan benar di kelas positif dan *Tingkat Positif Palsu* (FPR) yaitu probabilitas bahwa sampel negatif salah diprediksi di kelas positif. Metrik ini baik digunakan untk distribusi kelas yang seimbang misalnya pada dataset OEG. Namun ROC-AUC dapat memberikan kinerja yang kurang tepat pada dataset yang tidak seimbang seperti dataset CEG. Hal tersebut karena ROC menganalisis FPR, jumlah *False Positive* dibagi jumlah sample negatif. FPR dianggap baik jika nilai yang dihasilkan kecil, karena menunjukkan lebih sedikit kesalahan positif. Namun pada data yang tidak seimbang, FPR cenderung tetap pada nilai yang kecil karena banyaknya angka negatif (membuat penyebut menjadi besar).
   c. Ketiga mempertimbangkan nilai *sensitivity* dan *specificity*. ROC-AUC dapat menyebabkan kinerja yang kurang tepat pada dataset yang tidak seimbang, maka penting untuk melihat persentase kelas positif dan negatif yang benar diprediksi untuk memastikan bahwa model yang dipilih dapat memprediksi kelas dengan baik selain dengan melihat hasil dari PR-AUC.

V. PENUTUP

5.1. Simpulan

Berdasarkan penelitian yang telah dilakukan mengenai klasifikasi gen esensial pada *Drosophila melanogaster* dapat diambil kesimpulan sebagai berikut.

1. Penelitian ini mengimplementasikan metode *Gate Recurrent Unit* (GRU) dalam mengklasifikasikan gen esensial pada *Drosophila melanogaster* dengan skema pembagian data 80% *training* 20% *validation* dan 90% *training* 10% *validation.* Terdapat 3 arsitektur yang dibangun berdasarkan rujukan pada penelitian Vazhayil, R & KP (2018) dan beberapa modifikasi parameter model. Model di-*training* dengan dengan *epoch* 30, *optimizer* Adam, *learning rate* 0.001 dan *batch size* 128.
2. Hasil klasifikasi yang didapatkan untuk masing-masing dataset sebagai berikut.
    a. Hasil terbaik pada dataset OEG didapat dari skema dengan pembagian data yaitu 90% *training* dan 10% *validation* pada Arsitektur II, jumlah *neuron* 64 dengan *pre-padding*. Nilai yang didapat adalah 73% *sensitivity*, 72% *specificity*, 73% nilai ROC-AUC dan 78% nilai PR-AUC.
    b. Pada dataset CEG, terdapat proses *undersampling* untuk menyeimbangkan dataset dengan teknik *random undersampling*. Hasil klasifikasi yang paling baik didapat dari skema dengan pembagian data pelatihan 90% *training* 10% *validation* pada Arsitektur III, jumlah *neuron* 128 dengan *post-padding*. Nilai yang didapat untuk *sensitivity* adalah 72%, *specificity* 48%, nilai ROC-AUC 60% dan nilai PR-AUC 44%.

3. Hasil perbandingan dengan penelitian terdahulu menunjukkan bahwa pada penelitian ini memiliki hasil yang lebih rendah dari penelitian sebelumnya oleh Beder, et al, (2021). Hal ini berarti metode *Gated Recurrent Unit* (GRU) pada penelitian ini, belum cukup baik dalam mengklasifikasikan DNA pada *Drosophila melanogaster* dengan parameter yang digunakan.

5.2. Saran

Adapun saran yang diberikan pada penelitian ini adalah sebagai berikut.

1. Penelitian ini dapat dilanjutkan menggunakan metode klasifikasi lainnya seperti LSTM, BiLSTM, BiGRU untuk memperoleh hasil klasifikasi yang lebih baik sebagai bahan perbandingan.
2. Penelitian ini dapat dilanjutkan dengan mencoba menggunakan parameter lain dalam membangun model yang dapat meningkatkan hasil klasifikasi.
3. Penelitian ini dapat dilanjutkan dengan teknik *imbalanced* data lainnya, *oversampling* ataupun teknik *undersamping* yang lain untuk dataset CEG.

DAFTAR PUSTAKA

Adams, M. D., Celniker , S. E., Holt, R. A., Evans, C. A., Gocayne, J. D., Amanatides, P. G., et al. (2000). The Genome Sequence of Drosophila melanogaster. *Science*, 2184-2195.

Agrawal, N., Dasaradhi, P., Mohmmed, A., Malhotra, P., Bhatnagar, R., & Mukherjee, S. (2003). RNA interference: biology, mechanism, and applications. *Microbiol Mol Biol Rev*, 657-685.

Ahmad, S., Asghar, M. Z., Alotaibi, F. M., & Khan, S. (2020). Classification of Poetry Text Into the Emotional States Using Deep Learning Technique. *IEEE*, 73865-73878.

Aromolaran, O., Beder, T., Oswald, M., Oyelade, J., Adebiyi, E., & Koenig, R. (2020). Essential gene prediction in Drosophila melanogaster using machine learning approaches based on sequence and functional features. *Computational and Structural Biotechnology Journal* , 612-621.

Aromolaran, O., Oyelade, J., & Adebiyi, E. (2021). Performance evaluation of features for gene essentiality prediction. *Earth and Environmental Science* (pp. 1-14). Ota Nigeria: IOP Conference Series.

Baba, T., Ara, T., Hasegawa, M., Takai, Y., Okumura, Y., Baba, M., et al. (2006). Construction of Escherichia coli K-12 in-frame, single-gene knockout mutants: the Keio collection. *Molecular Systems Biology*, 1-11.

Beder, T., Aromolaran, O., Dönitz, J., Tapanelli, S., Adedeji, E. O., Adebiyi, E., et al. (2021). Identifying essential genes across eukaryotes by machine learning. *NAR Genomics and Bioinformatic*, 1-13.

Borror, D., Triplehorn, C., & Johnson, N. (1992). *Pengenalan Pelajaran Serangga.* (P. Penerjemah: Soetiyono, Trans.) Yogyakarta: UGM Press.

Campos, T. L., Korhonen, P. K., Hodmann, A., Gasser, R. B., & Young, N. D. (2020). Combined use of feature engineering and machine-learning to predict essential genes in Drosophila melanogaster. *NAR Genomics and Bioinformatics*, 1-12.

Cho, K., Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *In Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014).*

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *EMNLP 2014 - 2014 Conf. Empir. Methods Nat. Lang.*, 1724–1734.

Davis, J., & Goadrich, M. (2006). The Relationship Between Precision-Recall and ROC Curves. *Proceedings of the 23rd international conference on Machine learning* (pp. 233-240). Madison: ACM.

Erke, A., & Pattynama, P. (1998). Receiver operating characteristic (ROC) analysis: Basic principlesn and application in radiologi. *European Journal of Radiology*, 88-94.

Ethan, B. (2005). Drosophila, the golden bug, emerges as a tool for human genetics. *Nature Reviews*, 9-23.

Gerdes, S. Y., Scholle, M. D., Campbell, J. W., Balazsi, G., Ravasz, E., Daugherty, M. D., et al. (2003). Experimental Determination and System Level Analysis of Essential Genes in Escherichia coli MG1655. *JOURNAL OF BACTERIOLOGY*, 5674-5684.

Glick, B., Pasternak, J., & Patten, C. (2010). Molecular Biotechnology: Principles and Applications of Recombinant DNA. *DC* (pp. 117-118). Washington: ASM Press.

Gunasekaran, H., Ramalakshmi, K., Arokiaraj, A. R., Kanmani, S. D., Venkatesan, C., & Dhas, C. S. (2021). Analysis of DNA Sequence Classification Using CNN and Hybrid Models. *Hindawi Computational and Mathematical Methods in Medicine*, 1-12.

Guo, Y., Ju, Y., Chen, D., & Wang, L. (2021). Research on the Computational Prediction of Essential Genes. *Frontiers in Cell and Developmental Biology* , 1-9.

Hasanin, T., & Khoshgoftaar, T. (2018). The Effects of Random Undersampling with Simulated Class Imbalance for Big Data. *IEEE International Conference on Information Reuse and Integration for Data Science*, 70-79.

Kostadinov, S. (2017, Desember 16). *Understanding GRU Networks*. Retrieved Februari 20, 2023, from Towards Data Science: https://towardsdatascience.com/understanding-gru-networks-2ef37df6c9be

Liu, X. Y., Wu, J., & Zhou, Z. H. (2009). Exploratory Undersampling for Class-Imbalance Learning. *IEEE TRANSACTIONS ON SYSTEMS*, 539-550.

Nandi, S., Gangguli, P., & Sarkar, R. R. (2020). Essential gene prediction using limited gene essentiality information An integrative semi supervised machine learning strategy. *PLos One*, 1-29.

Peng, C., Lin, Y., Luo, H., & Gao, F. (2017). Resources to Identify and Predict Bacterial Essential Genes. *Frontiers in Cell and Developmental Biology* , 1-13.

Perveen, F. K. (2018). *Model for Recent Advances in Genetics and Therapeutics.* doi: 10.5772/67731.

Rai, A., & Borah, S. (2020). *Study of Various Methods for Tokenization.* Singapore.

Reddy, D. M., & Reddy, N. S. (2019). *EFFECTS OF PADDING ON LSTMS AND CNNS.* Retrieved from https://arxiv.org/pdf/1903.07288.pdf

Reznikoff , W. S., & Winterberg, K. M. (2008). Transposon-based strategies for the identification of essential bacterial genes. *Methods in Molecular Biology*, 13-26.

Rio, A. L.-d., Martin, M., Lluna, A. P., & Saidi, R. (2020). Efect of sequence padding on the performance of deep learning models in archaeal protein functional prediction. *Scientific Reports*.

Robert, J. (2005). *Genetic Analysis and Principles* (Third Edition ed.). McGraw Hill International edition.

Rogers, K. (2011). New Thinking about Genetics. New York: Britannica Educational Publishing.

Shadab, S., Khan, M. T., Neezi, N. A., Adilina, S., & Shatabda, S. (2020). DeepDBP: Deep neural networks for identification of DNA-binding proteins. *Informatics in Medicine Unlocked*, 1-7.

Sofaer, H., Hoeting, J., & Jarnevich, C. (2018). The area under the precision-recall curve as a performance metric for rare binary events. *British Ecological Society*, 565–577.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 1929-1958.

Tilaki, K. H. (2013). Receiver Operating Characteristic (ROC) Curve Analysis for Medical Diagnostic Test Evaluation. *Caspian J Intern Med*, 627–635.

Ullah, K., Rashad, A., Khan, M., Ghadi, Y., Aljuaid, H., & Nawaz, Z. (2022). A Deep Neural Network-Based Approach for Sentiment Analysis of Movie Reviews. *Hindawi*, 1-9.

Vazhayil, A., R, V., & KP, S. (2018, September 11). *DeepProteomics: Protein family classification using Shallow and Deep Networks.* Retrieved from https://arxiv.org/abs/1809.04461

Werdhana, R. W. (2017). *Klasifikasi Gen yang Terkait Sindrom Alzheimer Menggunakan Metode Naive Bayes Classifier, Binary Logistic Regression dan Logictis Regression Ensemble.* Surabaya: Institut Teknologi Sepuluh November.

Zhang, Z., & Ren, Q. (2015). Why are essential genes essential? – The essentiality of Saccharomyces genes. *Microbial Cell*, 280-287.