BAB I
PENDAHULUAN

1.1. Latar Belakang

Peningkatan pertumbuhan ekonomi adalah indikator tercapainya pembangunan ekonomi suatu negara. Manifestasi dari pertumbuhan ekonomi di negara tertentu tercermin dalam kesinambungan faktor-faktor ekonomi yang saling mempengaruhi dalam jangka panjang. Bagi negara berkembang seperti Indonesia, pembangunan ekonomi secara umum merupakan aspek kunci pembangunan yang bertujuan untuk meningkatkan pembangunan nasional dan berdampak pada sektor-sektor lainnya. Dalam upaya membangun ekonomi negara, Indonesia perlu memperhatikan pertumbuhan ekonomi, yang merupakan salah satu faktor penting dalam keberhasilan pembangunan ekonomi suatu negara. Peningkatan permintaan ekspor mengindikasikan peningkatan produksi domestik, yang pada gilirannya dapat memacu aktivitas ekonomi domestik sehingga menyebabkan pertumbuhan ekonomi dalam negeri. Tingkat keberhasilan pembangunan ekonomi suatu negara dapat dilihat dari seberapa besar pertumbuhan ekonomi yang terjadi dalam periode waktu tertentu. Pertumbuhan ekonomi sangat penting untuk keberlanjutan masyarakat dan negara. Bagi sebagian orang, pertumbuhan ekonomi memiliki dampak positif dan akan menghasilkan hasil yang baik bagi kesejahteraan masyarakat. Perdagangan internasional merupakan salah satu kegiatan ekonomi yang diharapkan mampu mendukung pembangunan nasional di mana dalam kegiatan perdagangan internasional terdapat kegiatan ekspor-impor (Tondolambung, et al., 2021).

Perekonomian Indonesia sangat bergantung pada sektor ekspor, termasuk ekspor non-migas yang meliputi berbagai produk pertanian, manufaktur, dan sumber daya alam lainnya. Ekspor non-migas memainkan peran krusial dalam menghasilkan devisa negara, menciptakan lapangan kerja, dan merangsang pertumbuhan ekonomi. Ekspor non migas memainkan peran penting dalam perekonomian Indonesia. Kontribusinya terhadap Produk Domestik Bruto (PDB) dan penciptaan lapangan kerja cukup signifikan. Pada tahun 2023, nilai ekspor non migas mencapai USD 242,90 miliar, atau sekitar 93.84% dari total ekspor Indonesia, jika dibandingkan dengan nilai ekspor non migas pada tahun 2022 dimana nilai ekspor non migas mencapai USD 275,96 miliar. Bisa kita lihat bahwa dari 2022 sampai 2023 turun sebanyak 11.33% (Badan Pusat Statistika Indonesia, 2024). Nilai ekspor non migas Indonesia mengalami fluktuasi dari tahun ke tahun. Fluktuasi ini dipengaruhi oleh berbagai faktor, seperti kondisi ekonomi global, harga komoditas, kebijakan pemerintah, dan kemampuan ekspor Indonesia.

Indonesia perlu meningkatkan ekspor negara untuk menumbuhkan perekonomian negara. Agar st*rate*gi pengembangan sektor perekonomian negara bisa tepat sasaran, diperlukan suatu prediksi untuk menentukan seberapa besar ekspor dan impor barang yang harus dijalankan negara. Dalam konteks global yang dinamis, kemampuan untuk memprediksi nilai ekspor non-migas dengan akurat menjadi sangat penting bagi pembuat kebijakan dan pelaku bisnis untuk merencanakan st*rate*gi dan mengambil keputusan yang tepat. Sebelumnya, Dhatu, et, al (2021) sudah melakukan penelitian untuk prediksi ekspor dan impor migas dengan metode *Extreme Learning Machine* (ELM)

Penelitian ini mendapatkan hasil dengan dataset ekspor didapatkan rata – rata nilai *Mean Absolute Percentage Error* (MAPE) terkecil sebesar 6,6742% untuk perbandingan jumlah data *training* : *testing* 70%:30%, jumlah fitur data 5, dan jumlah *hidden neuron* 8, sementara untuk dataset impor, hasil terbaik diperoleh dengan perbandingan jumlah data *training* : *testing* 80%:20%, jumlah fitur data 4, dan jumlah *hidden neuron* 10 dengan nilai rata – rata MAPE akhir yaitu 10,0515%.

Penelitian yang dilakukan oleh Prissy, et, al (2023) Peramalan Nilai Ekspor Migas di Indonesia dengan Model *Long Short Term Memory* (LSTM) dan *Gated Recurrent Unit* (GRU) dari hasil penelitian ini, diperoleh akurasi tertinggi dalam prediksi nilai ekspor migas menggunakan model terbaik LSTM dengan optimasi Nadam pada percobaan menggunakan nilai parameter α 0.001, jumlah *neuron* 20, *epoch* 100, dan nilai MAPE 12.8% dengan akurasi 87.2%.

Penelitian lainnya yang dilakukan oleh Emanuella, et, al (2022) Model Prediksi Harga Saham Apple Inc Pada Beberapa Bursa Efek Menggunakan Metode Multivariate Gated Recurrent Unit dari hasil penelitian ini, diperoleh evaluasi kinerja yang digunakan untuk mengukur tingkat kesalahan prediksi adalah MAE, RMSE, MAPE dan RMSPE. Berdasarkan hasil MAE dan RMSE dari AAPL(Nasdaq), APC.F(Frankfurt) dan AAPL.MX(Mexico) memberikan hasil tingkat kesalahan yang bernilai kecil. Adapun berdasarkan hasil evaluasi MAPE dan RMSPE memberikan hasil yang sangat baik dengan masing – masing persentase kesalahan yang dihasilkan < 10%.

Dalam konteks ini, penelitian ini bertujuan untuk mengembangkan model prediksi yang efektif untuk nilai ekspor non-migas di Indonesia. Dengan mengintegrasikan teknik pembelajaran mesin dan analisis data *time* series, penelitian ini akan fokus pada penggunaan arsitektur *Stacked Gated Recurrent Unit* (GRU), *Bidirectional* GRU, *Attention Based* GRU, dan *Seasonal Trend Decomposition using Loess* (STL) GRU untuk meramalkan nilai ekspor non-migas di masa depan yang berdasarkan data historis nilai ekspor non-migas.

1.2. Perumusan masalah

Berdasarkan uraian latar belakang, rumusan masalah pada penelitian ini sebagai berikut:

1. Bagaimana performa keempat model *Stacked* GRU, *Bidirectional* GRU, *Attention Based* - GRU, dan STL – GRU dalam memprediksi nilai ekspor non-migas di Indonesia?
2. Antara model *Stacked* GRU, *Bidirectional* GRU, *Attention Based* - GRU, dan STL – GRU, manakah yang menunjukkan hasil prediksi lebih akurat untuk *time* series univariate dari nilai ekspor non-migas di Indonesia?
3. Bagaimana melakukan *deployment* aplikasi website terhadap hasil keempat model yang telah dibuat dalam melakukan prediksi nilai ekspor non-migas di Indonesia?

Penelitian ini memiliki batasan-batasan permasalahan yang akan diteliti. Adapun batasan masalah tersebut adalah:

1. Data yang digunakan dalam penelitian ini adalah data historis bulanan jumlah nilai ekspor non-migas di Indonesia.

2. Metode yang digunakan dalam penelitian ini adalah metode *deep learning* dengan arsitektur jaringan syaraf tiruan *Stacked* GRU, model *Bidirectional* GRU, *Attention Based* – GRU, dan STL - GRU.
3. Analisis yang dilakukan hanya berdasarkan data historis bulanan jumlah nilai ekspor non-migas tanpa memperhitunkan faktor-faktor pendukung yang dapat menyebabkan perubahan jumlah nilai ekspor non-migas seperti tingkat inflasi, kurs nilai tukar, permintaan global, serta kejadian alam.

1.3. Tujuan dan Manfaat

Tujuan penelitian ini dilakukan adalah sebagai berikut:

1. Mengetahui kemampuan keempat model *Stacked* GRU, *Bidirectional* GRU, *Attention Based* - GRU, dan STL – GRU dalam memprediksi nilai ekspor non-migas di Indonesia.
2. Membandingkan keempat model tersebut untuk menentukan mana yang lebih baik dalam memprediksi nilai ekspor non-migas di Indonesia.
3. Melakukan *deployment* aplikasi website terhadap hasil keempat model *Stacked* GRU, model *Bidirectional* GRU, *Attention Based* – GRU, dan STL - GRU yang telah dibuat dalam melakukan prediksi nilai ekspor non-migas di Indonesia.

Penelitian ini diharapkan bermanfaat bagi akademisi, pemerintah, dan pengusaha. Hasil penelitian ini dapat menjadi referensi akademis yang memberikan kontribusi pada literatur ilmiah di bidang data science. Temuan penelitian ini dapat membantu pemerintah dalam merumuskan st*rate*gi perdagangan ekspor non-migas yang lebih efektif dan efisien. Hasil penelitian ini juga dapat membantu pengusaha dalam menganalisis tren pasar dan merencanakan st*rate*gi ekspor yang tepat untuk meningkatkan daya saing di pasar global.

1.4. Landasan Teori

1.4.1. Ekspor

Ekspor dapat diartikan sebagai pengiriman barang atau jasa ke luar negeri yang dilakukan oleh individu atau perusahaan dalam satu negara untuk dijual di negara lain. Ekspor merupakan komponen penting dalam neraca perdagangan dan menjadi sumber devisa bagi suatu negara. Ekspor dapat mendorong pertumbuhan ekonomi, menciptakan lapangan kerja, dan merangsang inovasi dalam produksi barang dan jasa (Krugman & Obstfeld, 2009). Menurut Amir M.S (2004), kegiatan ekspor dilakukan oleh suatu negara dengan tujuan :

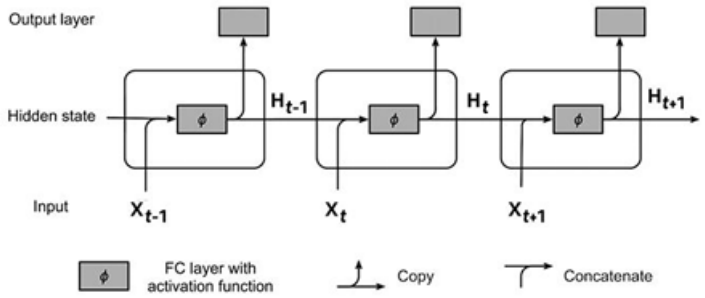
- Membuka pasar baru di luar negeri sebagai perluasan pasar domestik (membuka pasar ekspor). Sehingga dapat menciptakan iklim usaha dan ekonomi yang kondusif baik secara nasional maupun global.
- Memanfaatkan kelebihan kapasitas terpasang (idle capacity).
- Membiasakan diri bersaing dalam pasar internasional sehingga terlatih dalam persaingan yang ketat dan terhindar dari sebutan jago kandang.

1.4.2. Prediksi

Prediksi adalah proses memperkirakan sesuatu yang paling mungkin akan terbukti dengan membandingkan informasi yang dimiliki masa lalu dengan informasi yang dimiliki sekarang. Ini dapat berupa ramalan, proyeksi, atau perkiraan berdasarkan asumsi teoritik atau penilaian pakar tentang situasi masa depan. Dalam konteks ekonomi, prediksi dapat membantu dalam perencanaan dan pengambilan keputusan st*rate*gis. Model statistik dan komputasional, termasuk *machine learning* dan *deep learning*, telah banyak digunakan untuk membuat prediksi yang lebih akurat (Hyndman, et al., 2008). Peramalan (forecasting) merupakan bagian terpenting bagi setiap perusahaan ataupun organisasi bisnis dalam setiap pengambilan keputusan manajemen. Peramalan itu sendiri bisa menjadi dasar bagi perencanaan jangka pendek, menengah maupun jangka panjang suatu perusahaan. Di dalam sebuah peramalan (forecasting) dibutuhkan sedikit mungkin kesalahan di dalamnya. Agar dapat meminimalisir tingkat kesalahn tersebut, maka akan lebih baik jika peramalan tersebut dilakukan dalam suatu angka atau kuantitatif.

1.4.3. *Recurrent Neural Network*

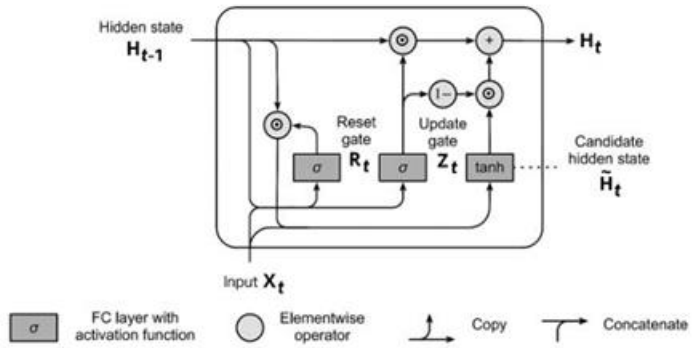
Recurrent Neural Network (RNN) merupakan salah satu model *neural network* yang mengakomodasi output jaringan untuk menjadi input jaringan kembali yang merupakan pengembangan dari feedforward *neural network* yang mampu memproses data sequential. *Recurrent Neural Network* (RNN) adalah kelas jaringan saraf yang dapat memproses sekuensial data dengan memanfaatkan koneksi internal yang memungkinkan informasi dari input sebelumnya untuk persisten. RNN sangat cocok untuk aplikasi seperti pemrosesan bahasa alami dan *time* series analysis (Goodfellow, et al., 2016). RNN adalah jaringan saraf berulang karena nilai *neuron* pada *hidden layer* sebelumnya digunakan kembali sebagai data input. Penggunaan *neuron* pada *hidden layer* akan disimpan ke dalam context *layer* (Wardana, 2020). Data pada context *layer* ini akan digunakan sebagai input pada *time step* berikutnya. Context *layer* menjadi sebuah memori berisi informasi dari setiap pemrosesan *time step* sebelumnya dan selanjutnya urutan data lain dapat dihasilkan setelah urutan data terdahulu dipelajari (Siringoringo, 2021). Arsitektur RNN dapat dilihat pada Gambar 1.



Gambar 1 Arsitektur Neural Network

1.4.4. *Gated Recurrent* Unit

Gated Recurrent Unit (GRU) merupakan salah satu varian dari RNN yang mengatasi beberapa keterbatasan model RNN standar, khususnya masalah vanishing gradient, dengan memperkenalkan struktur '*gate*' dalam arsitekturnya (Cho, et al., 2014). Tujuan utama dari pembuatan GRU adalah untuk membuat setiap *recurrent* unit untuk dapat menangkap dependencies dalam skala waktu yang berbeda-beda secara adaptif (Wardana, 2020). Serupa dengan LSTM, GRU juga menggunakan sistem gerbang, arsitektur GRU lebih sederhana daripada LSTM. GRU tidak menggunakan *cell state*, tetapi memanfaatkan *hidden state* untuk menyimpan informasi (Ghudafa, et al., 2022). Di dalam GRU, komponen pengatur alur informasi tersebut disebut sebagai *gate* dan GRU mempunyai 2 *gate*, yaitu *reset gate* dan *update gate*. Banyaknya informasi dari *time step* terdahulu yang dapat dilupakan ditentukan pada *reset gate*. Sementara *update gate* akan menentukan seberapa banyak informasi dari *time step* terdahulu yang dapat disimpan untuk digunakan sebagai input untuk *time step* berikutnya. Kemampuan GRU dirancang untuk menjadi lebih baik dari LSTM terutama untuk dataset yang jumlahnya sedikit (Hastomo, et al., 2021). Arsitektur GRU dapat dilihat pada Gambar 2.



Gambar 2 Arsitektur Gated Recurrent Unit

GRU merupakan *cell* yang mempunyai 2 *gate* dan 3 fungsi aktivasi, yaitu dua sigmoid dan sebuah tanh. Dengan *gate* dan fungsi aktivasi yang sedikit ini tentunya akan mempercepat proses pengolahan data yang umumnya berjumlah sangat besar. Fungsi aktivasi memutuskan apakah *neuron* harus diaktifkan atau tidak agar dapat berfungsi dengan benar dan memastikan bahwa *neural network* belajar untuk menggunakan informasi yang berguna. Dalam model GRU menggunakan dua jenis fungsi aktivasi yaitu sigmoid dan tanh. fungsi aktivasi sigmoid mentransformasi range nilai dari input menjadi antara 0 dan 1, fungsi sigmoid diuraikan pada persamaan 1.

$$f(x) = \sigma(x) = \frac{1}{(1 + e^{-x})} \quad (1)$$

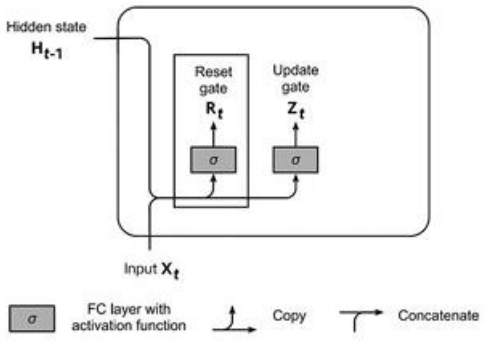
Sedangkan fungsi aktivasi tanh mentransformasi range nilai antara -1 sampai 1, fungsi tanh dapat diuraikan pada persamaan (2).

$$f(x) = tanh(x) = \frac{2}{(1 + e^{-2x})} - 1 \quad (2)$$

Di mana x = data input dan $e \approx 2.718281828459045\ldots$. Karakteristik terpusat di nol pada fungsi aktivasi tanh dapat lebih mudah memusatkan data sehingga proses pengoptimalan menjadi lebih mudah dibanding pada fungsi aktivasi sigmoid dengan threshold berada di 0.5 yang hanya dapat memutuskan bahwa input yang diberikan hanya memiliki dua tipe kelas. Karena output fungsi sigmoid yang tidak berpusat pada membuat proses optimal lebih sulit, sehingga fungsi tanh selalu lebih baik jika dibandingkan dengan fungsi sigmoid.

1.4.4.1. *Reset Gate*

Langkah pertama pada arsitektur GRU adalah dengan menentukan bagaimana menggabungkan informasi dari *time step* sebelumnya dan masukan baru di *reset gate*. Pada proses ini *reset gate* akan menentukan berapa banyak informasi dari *time step* terdahulu yang dapat dilupakan menggunakan fungsi aktivasi sigmoid. Output dari *reset gate* bernilai 0 dan 1. Jika output semakin mendekati 0 berarti informasi dari *time step* sebelumnya tidak terlalu berpengaruh dan akan dihapus sedangkan jika mendekati 1 berarti informasi dari *time step* terdahulu berpengaruh dan akan disimpan. Proses pada *reset gate* dapat dilihat pada Gambar 3.



Gambar 3 Reset Gate

Perhitungan pada *reset gate* diuraikan pada persamaan (3).

$$R_t = \sigma(X_t W_r + b_{xr} + H_{t-1} W_r + b_{hr}) \quad (3)$$

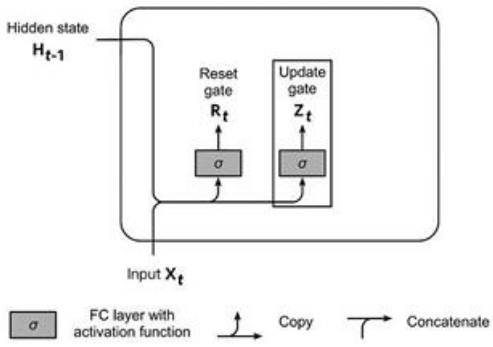
Keterangan:

R_t : *Reset gate*
σ : Fungsi *sigmoid*
X_t : Nilai data *input*
W_r : Nilai *weight* pada *reset gate*

b_{xr}, b_{hr} : Nilai bias pada *reset gate*
H_{t-1} : *Hidden state* dari *time step* sebelumnya

1.4.4.2. *Update Gate*

Langkah selanjutnya adalah menentukan seberapa banyak informasi dari *time step* terdahulu yang dapat disimpan untuk perhitungan *hidden state* yang digunakan sebagai input pada *time step* berikutnya dan menentukan pengaruh informasi dari *time step* terdahulu pada output di *time step* saat ini. Proses ini berlangsung di *update gate* menggunakan fungsi aktivasi sigmoid. Ketika output sama dengan 1, informasi dari *time step* sebelumnya memiliki pengaruh pada output saat ini, dan jika output sama dengan 0 maka informasi dari *time step* sebelumnya tidak berpengaruh pada output saat ini. Proses pada *update gate* dapat dilihat pada Gambar 4.



Gambar 4 Update Gate

Perhitungan pada *update gate* diuraikan pada persamaan (4)

$$Z_t = \sigma(X_t W_z + b_{xz} + H_{t-1} W_z + b_{hz}) \quad (4)$$

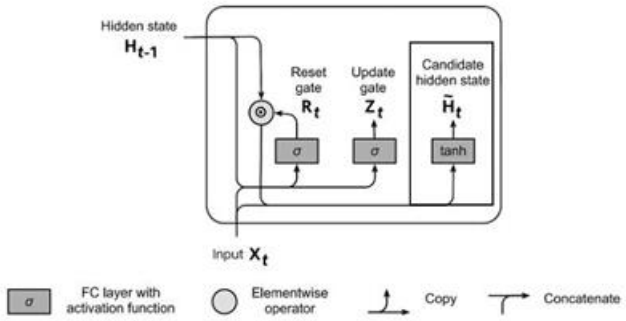
Keterangan:

Z_t : *Update gate*
σ : Fungsi *sigmoid*
X_t : Nilai data *input*
W_z : Nilai *weight* pada *update gate*
b_{xz}, b_{hz} : Nilai bias pada *update gate*
H_{t-1} : *Hidden state* dari *time step* sebelumnya

1.4.4.3. *Candidate Hidden State*

Langkah selanjutnya adalah menentukan *candidate hidden state* atau implicit output pada *time step* saat ini (t) dari informasi yang relevan pada *time step* masa lalu (t-1) dengan menggunakan fungsi aktivasi tanh. *candidate hidden state* ini bukanlah output final dari unit melainkan content memory pada *time step* saat ini. Nilai *candidate hidden*

state dipengaruhi oleh output dari *reset gate*. Proses penentuan nilai *candidate hidden state* dapat dilihat pada Gambar 5.



Gambar 5 Candidate Hidden State

Proses penentuan *candidate hidden state* diuraikan pada persamaan (5).

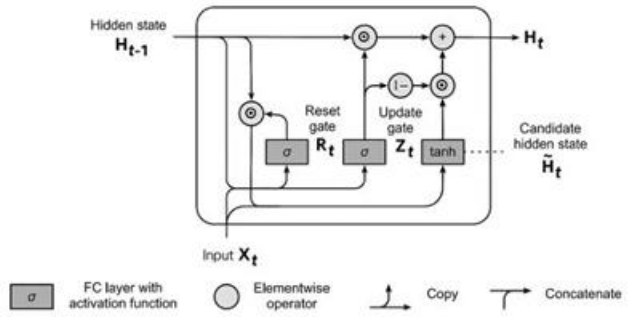
$$\tilde{H}_t = tanh(X_t W_h + b_{xh} + R_t \odot (H_{t-1} W_h + b_{hh})) \quad (5)$$

Keterangan:

$\tilde{H}_t$	: *Candidate hidden state*
R_t	: Ouput pada *reset gate*
X_t	: Nilai data *input*
W_h	: Nilai *weight* pada *candidate hidden state*
b_{xh}, b_{hh}	: Nilai bias pada *candidate hidden state*
H_{t-1}	: *Hidden state* dari *time step* sebelumnya

1.4.4.4. *Hidden State*

Proses terakhir yaitu jaringan akan menghitung output akhir dari unit saat ini dan meneruskannya ke *time step* berikutnya sebagai *hidden state*. *Hidden state* yang akan dikirimkan ke *time step* berikutnya akan digunakan untuk menghitung kembali output pada unit *time step* tersebut. Prosesnya akan berjalan sama dan berulang seperti pada proses di *time step* saat ini hanya dengan nilai input yang berbeda. Perhitungan output terakhir ini dipengaruhi oleh nilai *candidate hidden state*, nilai *hidden state* pada *time step* sebelumnya dan output dari *update gate*. Proses penghitungan nilai *hidden state* dapat dilihat pada Gambar 6.



Gambar 6 Output Hidden State

Proses penghitungan output terakhir diuraikan pada persamaan (6).

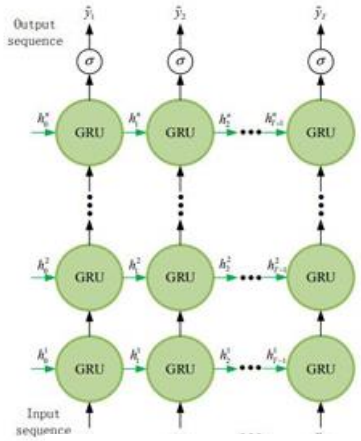
Keterangan:

$$H_t = Z_t \odot H_{t-1} + (1 - Z_t) \odot \tilde{H}_t \quad (6)$$

H_t : *Output*
$\tilde{H}_t$: *Candidate hidden state*
Z_t : *Output* pada *update gate*
H_{t-1} : *Hidden state* dari *time step* sebelumnya

1.4.5. *Stacked Gated Recurrent* Unit

Stacked GRU adalah variasi dari *Gated Recurrent Unit* (GRU), yang merupakan salah satu jenis arsitektur dalam pembelajaran mesin untuk pemrosesan urutan data, seperti teks atau rangkaian waktu. Seperti namanya, "*stacked*" (bertumpuk) mengacu pada penumpukan beberapa lapisan GRU satu di atas yang lain, yang bertujuan untuk menangkap abstraksi pada level yang lebih tinggi dalam data yang diproses.



Gambar 7 Struktur Stacked GRU

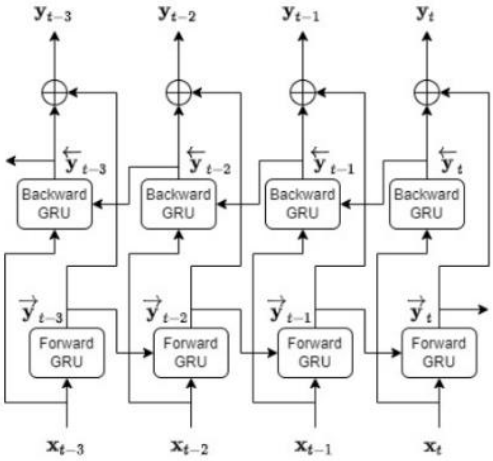
Input dari setiap unit GRU di tengah adalah Output dari lapisan tersembunyi unit GRU lapisan atas, bisa kita lihat pada persamaan (7):

$$\begin{cases} Z_t^i = \sigma\left(W_z^i \cdot \left[H_{t-1}^i, X_t^{i-1}\right]\right) \\ R_t^i = \sigma\left(W_z^i \cdot \left[H_{t-1}^i, X_t^{i-1}\right]\right) \\ \widetilde{H}_t^i = tanh\left(W_h^i \cdot \left[R_t^i \odot H_{t-1}^i, X_t^{i-1}\right]\right) \\ H_t^i = Z_t^i \odot H_{t-1}^i + \left(1 - Z_t^i\right) \odot X_t^{i-1} \end{cases} \quad (7)$$

1.4.6. *Bidirectional Gated Recurrent* Unit

Bi-GRU (*Bidirectional Gated Recurrent* Unit) adalah jenis arsitektur jaringan saraf berulang (RNN) yang digunakan untuk memproses data sekuensial seperti teks, suara, dan deret waktu. Bi-GRU mampu menangkap informasi dari kedua arah sekuensial, baik dari awal hingga akhir maupun dari akhir hingga awal, sehingga dapat meningkatkan kinerja model dalam tugas-tugas pemrosesan bahasa alami (NLP), pengenalan ucapan, dan analisis deret waktu.

Bi-GRU terdiri dari dua unit GRU yang berjalan secara paralel: satu unit yang memproses sekuensial dari awal hingga akhir (forward GRU) dan satu unit yang memproses sekuensial dari akhir hingga awal (backward GRU). Kedua unit ini kemudian menggabungkan hasil pemrosesannya untuk menghasilkan representasi vektor akhir yang menggabungkan informasi dari kedua arah sekuensial.



Gambar 8 Strukur Bidirectional GRU

Lapisan GRU ke depan dijelaskan oleh persamaan berikut:

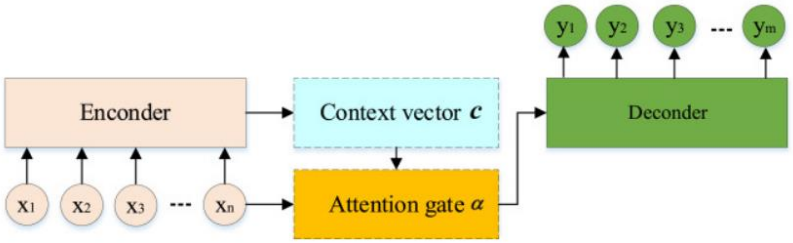
$$\begin{aligned} \vec{y}_t &= (1 - z_t) \odot \vec{y}_{t-1} + z_t \odot h_t \\ h_t &= tanh\left(W_h x_t + U_h\left[r_t \odot \vec{y}_{t-1}\right] + b_h\right) \\ z_t &= \sigma\left(W_z x_t + U_z \vec{y}_{t-1} + b_z\right) \\ r_t &= \sigma\left(W_r x_t + U_r \vec{y}_{t-1} + b_r\right) \end{aligned} \quad (8)$$

di mana z adalah vektor gerbang pembaruan, r adalah vektor gerbang *reset*, h adalah output dari lapisan tersembunyi, $W_{h,z,r}$, $U_{h,z,r}$ adalah matriks bobot, $B_{h,z,r}$ adalah vektor bias, ⊙ adaah *Element-wise vector multiplication*, dan σ adalah fungsi sigmoid logistik. Persamaan yang sama berlaku untuk lapisan GRU mundur kecuali bahwa indeks t-1 diganti dengan t+1. Keluaran akhir dari lapisan Bi-GRU adalah

$$y_t = \vec{y}_t + \overleftarrow{y}_t$$

1.4.7. Mekanisme *Attention*

Mekanisme attention adalah suatu teknik dalam deep learning yang memungkinkan model untuk fokus pada bagian-bagian penting dari input data. Ini membantu model untuk memproses informasi secara lebih efisien dan efektif. Seperti halnya opada tak manusia akan fokus pada beberapa area penting untuk mendapatkan informasi yang berguna. Berdasarkan prinsip ini (V. Mnih, et al., 2014) mengusulkan model perhatian siklik berdasarkan mekanisme perhatian dari fitur-fitur penambangan yang penting untuk terjemahan mesin. Mekanisme perhatian telah banyak digunakan dalam pemrosesan data deret waktu (A.-Y. Tang, et al., 2014).



Gambar 9 Struktur Attention Mekanism

Attention mekanism digunakan sebagai fungsi gerbang. Distribusi bersyarat $P(y_t|y_{t-1}, y_{t-2}, \dots, c) = g\left(h_t, y_{t-1}, (\alpha_t \cdot \bar{h})\right)$ dari t setiap saat tidak diwakili oleh keadaan tersembunyi terakhir dari RNN encoder tetapi oleh hasil kali titik dari gerbang perhatian α_t dan encoder output $\bar{h}$, bisa dilihat pada persamaan (9).

$$P(y_t|y_{t-1}, y_{t-2}, \dots, c) = g\left(h_t, y_{t-1}, (\alpha_t \cdot \bar{h})\right) \quad (9)$$

Attention Gate α_t dinyatakan sebagai:

$$\alpha_t = \frac{exp\big(a(x_t, q)\big)}{\sum_{T=1}^{k} exp\big(a(x_T, q)\big)} \quad (10)$$

Dimana q adalah vektor kueri terkait tugas. $a(x_t, q)$ adalah fungsi skor perhatian aditif yang rumusnya adalah

$$a(x_t, q) = V^T \tanh\left(W x_t + U_q\right) \quad (11)$$

Dimana U, V dan W adalah parameter jaringan yang diperoleh dengan melatih jaringan saraf.

1.4.8. *Seasonal Trend Decomposed* (STL)

Selain karakteristik tren yang paling umum, perubahan nlai ekspor melibatkan musiman dan sisa tertentu, yang disebabkan oleh tindakan gabungan dari berbagai faktor yang mempengaruhi. Secara khusus, perubahan nilai ekspor menunjukkan karakteristik tren, musiman, dan sisa waktu berikut ini.

- Tren. Tren mengacu pada perubahan jangka panjang termasuk peningkatan, penurunan, atau stabilitas. Hal ini merupakan dasar fundamental untuk menilai keseluruhan pola perubahan kondisi di masa depan.
- Musim. Musim mengacu pada pola berulang dalam suatu periode, biasanya dalam satu tahun. Hal ini berkontribusi pada pemahaman yang lebih akurat mengenai fitur statistik tertentu yang signifikan dalam periode waktu tertentu.
- Sisa. Sisa mengacu pada komponen yang mengandung ketidakpastian atau keacakan, yang sulit untuk diungkapkan secara langsung oleh model analitis matematis. Tingkat fluktuasi sisa akan mempengaruhi akurasi prediksi kejadian secara keseluruhan.

Metode STL pada awalnya diusulkan oleh Cleveland dkk (Robert B. Cleveland, et al., 1990)., yang menerapkan regresi terbobot lokal yang kuat untuk memecah deret waktu menjadi: (1) *trend-cycle terms* (Tt), (2) *seasonal terms* (St), dan (3) *remainder terms* (Rt), seperti pada rumus (10).

$$Y_t = f(T_t S_t R_t), t \in (0, n] \quad (12)$$

Di mana Y_t adalah sampel input asli pada waktu t, dan n adalah panjang urutan. $f(\cdot)$ dapat diekspresikan dengan model aditif atau model perkalian. Dari jumlah tersebut, model aditif memungkinkan untuk menangkap karakteristik temporal urutan dengan periodisitas yang signifikan dengan lebih baik.

STL cocok dengan regresi polinomial dengan mengekstraksi setiap komponen lokal, dan dapat diterapkan untuk menganalisis semua jenis tren musiman (Bergmeir C, et al., 2016) (He, et al., 2021). Dibandingkan dengan teknik dekomposisi tradisional secara teoritis. STL didasarkan pada regresi berbobot lokal (*Loess*) dan menggunakan struktur siklus ganda.

- *Loess*. *Loess* adalah teknik nonparametrik untuk memperhalus perubahan tren dan komponen musiman, dan selanjutnya menghitung permukaan regresi lokal yang telah dipasangkan melalui pembobotan ulang secara berulang untuk memastikan pemasangan yang kuat.
- *Cycling steps*, STL terdiri dari siklus internal dan eksternal. Siklus internal digunakan untuk memperbarui komponen musiman dan tren Nilai ekspor, sedangkan siklus eksternal digunakan untuk mengatur ketahanan pada siklus internal berikutnya.

1.4.9. Inisialisasi *Hyperparameter*

Dalam pembelajaran mesin, *hyperparameter* adalah parameter yang menentukan detail proses pembelajaran. Hal ini berbeda dengan parameter yang menentukan model itu sendiri. *Hyperparameter* dapat diklasifikasikan sebagai *hyperparameter* model, yang biasanya tidak dapat disimpulkan saat memasang mesin ke set pelatihan, dan *hyperparameter* algoritma, yang ditetapkan sebelum memperkirakan parameter melalui algoritma pelatihan. Contoh *hyperparameter* model mencakup topologi dan ukuran jaringan saraf, sedangkan contoh *hyperparameter* algoritme mencakup *Learning rate* dan *Batch size*. Pilihan *hyperparameter* dapat memengaruhi performa model, serta waktu yang diperlukan untuk melatih dan menguji model. Beberapa *hyperparameter* tidak dapat dipelajari menggunakan metode pengoptimalan berbasis gradien dan harus disetel secara manual. Tunabilitas suatu algoritma, *hyperparameter*, atau *hyperparameter* yang berinteraksi adalah ukuran seberapa besar performa yang dapat diperoleh dengan menyetelnya. inisialisasi *hyperparameter* dalam penelitian ini meliputi *optimizer* dan *learning rate*, *batch size*, jumlah *epoch*, jumlah *hidden layer* dan jumlah *neuron* pada setiap *layer*.

1.4.10. GridSearch

GridSearch merupakan salah satu metode dalam pembelajaran mesin dan *deep learning* untuk menyetel *hyperparameter* dari sebuah model. *Hyperparameter* adalah pengaturan konfigurasi yang ditetapkan sebelum proses pelatihan dimulai (seperti laju pembelajaran, parameter regularisasi, atau jumlah lapisan dalam jaringan saraf). Tujuan dari GridSearch adalah untuk menemukan kombinasi optimal dari hyperparameter ini untuk memaksimalkan kinerja model. GridSearch dilakukan dengan pendekatan brute-force untuk menemukan hiperparameter optimal untuk sebuah model dengan mencoba setiap kombinasi yang mungkin, dan sering digunakan ketika sumber daya komputasi memungkinkan dan ruang hiperparameter dapat dikelola. GridSearch menyediakan cara sistematis untuk mengeksplorasi hiperparameter dalam machine learning dan deep learning. Meskipun intensif secara komputasi, ini adalah metode yang mudah dan menyeluruh untuk memastikan penyetelan model yang optimal, terutama ketika sumber daya dan ukuran grid dapat dikelola.

1.4.11. *Optimizer* dan *Learning Rate*

Optimizer adalah algoritme yang memperbarui parameter model untuk meminimalkan fungsi kerugian. Ini adalah komponen penting dari proses pelatihan dalam pembelajaran mesin dan pembelajaran mendalam. Pilihan pengoptimal dapat berdampak signifikan terhadap performa model. dalam penelitian ini menggunakan *Adaptive Moment Estimation* (Adam). Adam adalah algoritma optimasi *Learning rate* adaptif yang menggabungkan ide Adagrad dan *moment*um. Ini menggunakan *Learning rate* adaptif untuk setiap parameter dan rata-rata pergerakan gradien untuk mempercepat konvergensi.

Learning rate adalah *hyperparameter* yang mengontrol ukuran langkah algoritma pengoptimalan yang digunakan untuk melatih model. Ini menentukan seberapa banyak parameter model diperbarui dalam setiap iterasi proses optimasi. *Learning rate* yang lebih tinggi berarti pembaruan yang lebih besar, sedangkan *Learning rate* yang lebih rendah berarti pembaruan yang lebih kecil. Pilihan kecepatan pemelajaran dapat sangat memengaruhi performa model. Jika kecepatan pemelajaran terlalu tinggi, model akan terkonvergensi terlalu cepat ke solusi suboptimal, sedangkan jika kecepatan pemelajaran terlalu rendah, model mungkin terjebak dalam minimum lokal dan memerlukan lebih banyak iterasi agar dapat konvergen ke solusi optimal.

1.4.12. *Dropout*

Dropout adalah teknik regularisasi yang digunakan dalam jaringan *neural* untuk mencegah overfitting dan meningkatkan performa generalisasi model. Ia bekerja dengan "menghilangkan" secara acak (mengatur ke nol) persentase tertentu dari input dan unit tersembunyi selama proses pelatihan. Hal ini membantu model melakukan generalisasi dengan lebih baik dan mengurangi ketergantungan pada unit tertentu, menjadikan model lebih kuat dan kecil kemungkinannya untuk menyesuaikan dengan data pelatihan. Dengan melakukan drop pada *neuron* berarti, *neuron* yang dibuang tidak akan mengambil bagian dalam propagasi maju dan akan diberhentikan sementara dan bobot baru juga tidak diterapkan pada *neuron* pada saat melakukan propagasi mundur selama proses *training*.

1.4.13. Regularization

Regularisasi dapat didefinisikan sebagai modifikasi atau perubahan apa pun dalam algoritme pembelajaran yang membantu mengurangi kesalahan pada dataset pengujian, yang umumnya dikenal sebagai kesalahan generalisasi, tetapi tidak pada dataset yang disediakan atau dataset pelatihan. Model Regresi yang menggunakan regularisasi L2 disebut Regresi Ridge. Regularisasi ini menambahkan penalti ketika kompleksitas model meningkat. Parameter regularisasi (lambda) memberikan penalti pada semua parameter kecuali intersep sehingga model dapat menggeneralisasi data dan tidak overfit.

1.4.14. Normalisasi

Normalisasi data adalah proses pengorganisasian data agar tampak serupa di seluruh catatan dan bidang, yang meningkatkan kohesi jenis entri dan menghilangkan data tidak terstruktur dan redundansi. Ini merupakan langkah penting dalam mempersiapkan data untuk algoritma pembelajaran mesin, karena memastikan bahwa data berada dalam format yang konsisten sehingga dapat dengan mudah diproses oleh algoritma. Tujuan dari normalisasi data adalah menghindari fitur yang memiliki nilai yang lebih besar mendominasi fitur yang memiliki nilai lebih kecil (Wardana, 2020). Metode normalisasi data yang digunakan dalam penelitian ini adalah Min-Max Scaling Normalization. Persamaan umum untuk scaling dalam rentang [0, 1] diuraikan pada persamaan (13).

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (13)$$

Keterangan:

x' : Nilai hasil normalisasi
x : Nilai aktual
$x_{max}\ dan\ x_{min}$: Nilai tertinggi dan terendah dari data

1.4.15. Metode Evaluasi dan Kinerja Model

Melakukan evaluasi kinerja penting untuk pembelajaran mesin dan teknik pembelajaran mendalam. Hal ini dilakukan dengan tujuan untuk mengukur tingkat akurasi kinerja metode yang digunakan. Dalam penelitian ini kinerja diukur dengan fungsi kerugian yang menghitung *mean absolute error*, *mean absolute* persen *error*, dan *root mean square error*.

1.4.16. *Mean Absolute Error* (MAE)

Mean Absolute Error (MAE) adalah ukuran rata-rata kesalahan dalam kumpulan prediksi, tanpa memperhitungkan arahnya. Ini dihitung sebagai perbedaan absolut rata-rata antara nilai prediksi dan nilai sebenarnya dan digunakan untuk menilai efektivitas model regresi. Rumus MAE bisa dilihat pada persamaan (14).

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right| \quad (14)$$

Keterangan:

n : Jumlah data
$\hat{y}_i$: Nilai prediksi
y_i : Nilai asli

1.4.17. *Mean Absolute Percentage Error* (MAPE)

Mean Absolute Percentage Error (MAPE) adalah ukuran keakuratan sistem ramalan. Ini dihitung sebagai rata-rata nilai absolut persentase kesalahan antara nilai aktual dan perkiraan. MAPE dinyatakan dalam persentase, sehingga lebih mudah untuk dipahami dan dikomunikasikan. Hal ini sering digunakan dalam penganggaran dan peramalan, serta dalam analisis regresi dan evaluasi model. Rumus MAPE bisa dilihat pada persamaan (15).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right| \times 100\% \quad (15)$$

Keterangan:

n : Jumlah data
$\hat{y}_i$: Nilai prediksi
y_i : Nilai asli

1.4.18. *Root Mean Square Error* (RMSE)

Root Mean Square Error (RMSE) adalah besarnya tingkat kesalahan hasil prediksi. RMSE dikatakan akurat jika hasilnya semakin kecil atau mendekati 0. RMSE adalah hasil akar kuadrat MSE. Keakuratan metode estimasi kesalahan pengukuran ditandai dengan nilai RMSE yang kecil. Rumus RMSE bisa dilihat pada persamaan (16).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2} \quad (16)$$

Keterangan:

n : Jumlah data
$\hat{y}_i$: Nilai prediksi
y_i : Nilai asli

1.4.19. *R-squared* (R^2)

R-squared (R^2) adalah metrik evaluasi yang umum digunakan dalam tugas-tugas regresi, termasuk peramalan deret waktu. Ini mewakili proporsi varians dalam variabel dependen (target) yang dapat diprediksi dari variabel independen. juga dikenal sebagai koefisien determinasi, mengukur seberapa baik prediksi model mendekati titik data aktual. Koefisien determinasi mengukur fraksi variabilitas dalam variabel target yang dijelaskan oleh model.

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \quad (17)$$

Keterangan:

SS_{res} : Jumlah kuadrat sisa
SS_{tot} : Nilai kuadrat total
y_i : Nilai asli

1.4.20. *Deployment Model*

Deployment model dalam pembelajaran mesin mengacu pada proses pengintegrasian model pembelajaran mesin terlatih ke dalam lingkungan produksi yang dapat digunakan untuk membuat prediksi pada data baru secara *real-time*. Ini adalah tahap akhir dari proses pembelajaran mesin, saat model diterapkan untuk memecahkan masalah bisnis tertentu atau memenuhi persyaratan tertentu. Streamlit adalah sebuah framework berbasis python dan bersifat open-source yang dibuat untuk memudahkan dalam membangun aplikasi web di bidang data science dan *machine learning* yang interaktif (Singh, 2021). Dengan menggunakan framework ini hanya perlu install *library* Streamlit dan membuat file Python dalam merancang website sehingga tidak perlu lagi membuat file HTML, CSS, JavaScript, Flask, Django atau tools lainnya.

BAB II
METODE PENELITIAN

2.1. Waktu dan Lokasi Penelitian

Penelitian ini dilakukan di kediaman pribadi peneliti. Penelitian ini akan dilaksanakan mulai Mei 2024 sampai dengan September 2024, sedangkan penulisan tugas akhir dimulai pada Mei 2024 sampai dengan September 2024.

2.2. Dataset

Dalam penelitian ini, Data yang akan digunakan adalah data nilai non-ekspor perbulan di Indonesia dari Januari1993 hingga Juni 2024 dalam nilai juta US$. Data tersebut diperoleh dari website Badan Pusat Statistika Indonesia (BPS), yang merupakan sumber terpercaya untuk data terkait nilai non-ekspor di Indonesia. Ekstensi data ini adalah *comma separated values* (.csv) yang terdiri dari field periode dan nilai non-ekspor.

2.3. Instrumen Penelitian

Instrumen penelitian yang digunakan yaitu perangkat keras (*hardware*) dan perangkat lunak (*software*). Komponen utama dari penelitian ini adalah perangkat lunak (*software*). Penggunaan perangkat keras pada penelitian yaitu laptop dan smartphone. Berikut spesifikasi laptop yang digunakan pada penelitian kali ini:

Tabel 1. spesifikasi perangkat keras

No.	Perangkat Keras	Spesifikasi
1.	Operationg System (OS)	Windows 10
2.	Processor	Intel(R) Core(TM) i7
3.	RAM	16,00 GB
4.	System type	64-bit operating processor
5.	GPU	Intel(R) UHD Graphics

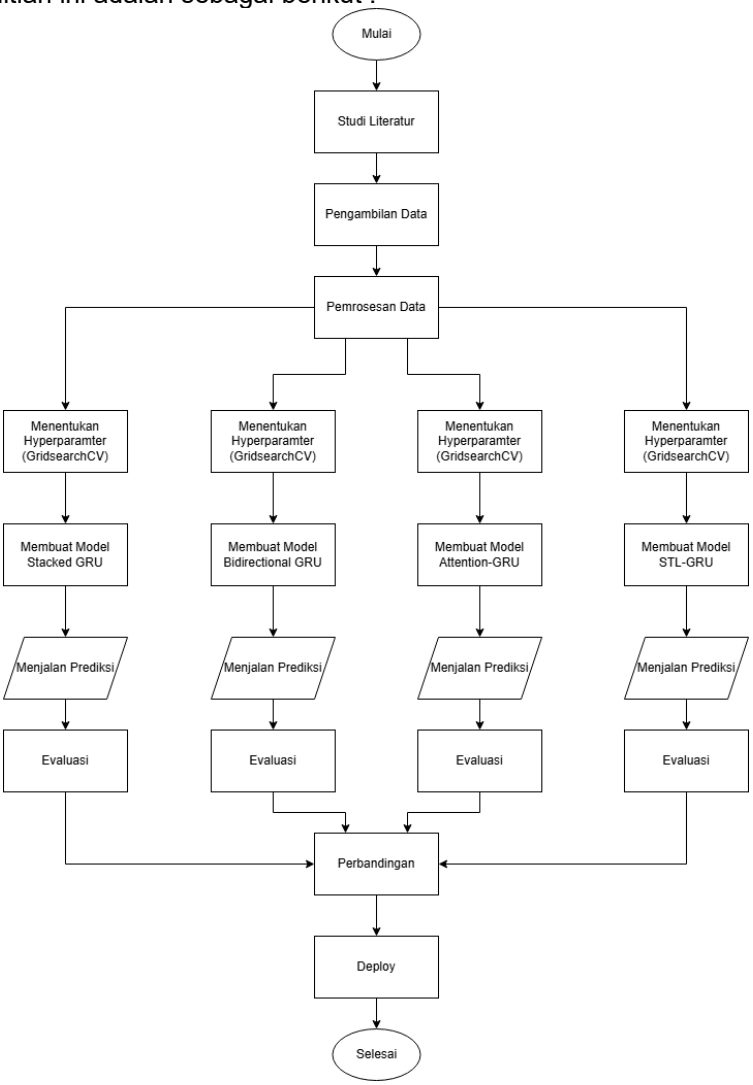
Berikut Perangkat Lunak (*Software*) yang digunakan selama penelitian di kerjakan:

Tabel 2. spesifikasi perangkat lunak

No.	Perangkat Lunak	Spesifikasi
1.	Google Colab	Digunakan untuk melakukan pengkodingan prediksi
2.	Github Codespace	Digunakan untuk melakukan pengkodingan untuk bagian webnya
3.	Draw.io	Digunakan untuk membuat diagram penelitian
4.	Streamlit	Dijadikan tempat untuk men*deploy* model
5.	Web Browser	Digunakan untuk mengakses Google Colab, Github Codespace, dan Draw.io
6.	Excel 2020	Digunakan dalam pemrosesan data dan analisis awal.

2.4. Diagram Alur Penelitian

Sub bab ini membahas tentang tahapan penelitian yang dilakukan mulai dari studi literatur, mempersiapkan data, mengimplementasi dan mengevaluasi model serta interpretasi dari hasil penelitian untuk mendapatkan kesimpulan yang baik. Peneliti menggunakan bahasa pemrograman python dengan bantuan beberapa modul diantaranya tensorflow, keras, pandas, numpy, dan matplotlib. Adapun tahapan tahapan dalam penelitian ini adalah sebagai berikut :



Gambar 10 Diagram Alur Penelitian

2.4.1.Studi Literatur

Pada tahap ini dilakukan pengumpulan literatur berupa konsep maupun teori yang berhubungan dengan masalah-masalah yang diangkat di tugas akhir ini. Sumber literatur dapat berupa jurnal, buku, majalah, dan sumber-sumber lain yang valid dan dapat dipercaya.

2.4.2.Pengambilan Data

Pada tahap ini dilakukan pengambilan data nilai non-ekspor perbulan di Indonesia dari tahun 1993 hingga 2023 dalam nilai juta USD($), yang diperoleh dari website Badan Pusat Statistika Indonesia (BPS).

2.4.3.Pemrosesan Data

Pada tahap ini data yang telah diperoleh akan melalui serangkaian langkah yang dilakukan untuk mengubah data mentah menjadi informasi yang berguna. Proses ini melibatkan berbagai teknik dan algoritma untuk membersihkan, mengolah, dan menganalisis data.

2.4.4.Menentukan *Hyperparameter*

Pada tahap Menentukan *Hyperparameter*, kita memasuki bagian penentuan Hyperparameter dengan menggunakan **gridsearchCV** dari proses pengembangan model *machine learning*. Setelah data kita bersihkan dan siap, langkah selanjutnya adalah membangun model yang mampu belajar dari data tersebut.

2.4.5. Membuat Model

Pada tahap ini setelah didapatkan hyperparameter yang optimal untuk masing-masing model, maka selanjutnya adalah dengan membangun ke empat model yaitu *stacked* GRU, *bidirectional* GRU, *attention mechanism*-GRU, dan STL-GRU.

2.4.6.Evaluasi

Pada tahap ini Setelah model dilatih, kita perlu mengevaluasi seberapa baik model tersebut dalam melakukan prediksi pada data yang belum pernah dilihat sebelumnya.

2.4.7.Perbandingan

Pada tahap ini kita membandingkan kinerja dari beberapa model yang telah kita latih. Tujuan utama dari perbandingan ini adalah untuk memilih model terbaik yang akan digunakan untuk memprediksi data baru.

2.4.8.*Deploy*

Pada tahap *deploy* dalam pengembangan model *machine learning* adalah langkah terakhir di mana model yang telah dilatih dan diuji diterapkan ke dalam lingkungan produksi untuk digunakan dalam aplikasi nyata, peneliti menggunakan *streamlit* sebagai tempat untuk men*deploy*.