BAB I
PENDAHULUAN

1.1 Latar Belakang

Data time series merupakan representasi dari fenomena yang diamati secara berurutan terhadap waktu. Bentuk data ini bersifat longitudinal dan sering digunakan dalam bidang keuangan, kesehatan, meteorologi, dan energi karena kemampuannya merekam dinamika suatu sistem dari waktu ke waktu (Cryer & Chan, 2008) . Dalam praktiknya, data time series memiliki pola keterkaitan antar waktu (autokorelasi) baik dalam jangka pendek maupun jangka panjang. Sifat keterikatan ini berbeda dari data *cross-sectional* dan menjadikan data *time series* memiliki tantangan tersendiri dalam hal pemodelan dan analisis (Wang et al., 2020). Ketika diaplikasikan dalam dunia keuangan, khususnya untuk prediksi harga saham, data time series memperlihatkan fluktuasi yang tidak hanya kompleks tetapi juga sangat dipengaruhi oleh faktor eksternal yang dinamis.

Model-model statistik klasik seperti *Autoregressive Integrated Moving Average* (ARIMA) dan *Generalized Autoregressive Conditional Heteroskedasticity* (GARCH) adalah model yang populer digunakan dalam analisis time series selama ini. ARIMA digunakan untuk data linier dan stasioner, sedangkan GARCH lebih banyak digunakan untuk memodelkan volatilitas pasar keuangan (Box et al., 2016; Durham, 2007). Beberapa penelitian telah menunjukkan efektivitas pendekatan ini dalam peramalan jangka pendek, seperti prediksi AOD (Taneja et al., 2016), prediksi muka air tanah (Gibrilla et al., 2018), serta estimasi penyebaran COVID-19 (Alzahrani et al., 2020; Singh et al., 2020; Sun, 2021). Namun, keterbatasan asumsi linieritas, stasioneritas, dan sensitivitas terhadap parameter menjadikan model ini kurang fleksibel untuk menangani pola nonlinier dan dinamika pasar saham yang cepat berubah. Sebagai respon atas keterbatasan tersebut, pendekatan berbasis *Machine Learning* (ML) diperkenalkan untuk menangani kompleksitas yang tidak dapat dijangkau oleh metode statistik tradisional.

Berbagai algoritma ML seperti *Support Vector Machine* (SVM), *Random Forest*, dan *Artificial Neural Network* (ANN) telah diaplikasikan dalam peramalan pasar saham (Choi & Kim, 2021; Escribano & Wang, 2021; Huang et al., 2005). Algoritma ini mampu menangkap pola nonlinier dan interaksi kompleks antar variabel. Namun, ML klasik memiliki kelemahan dalam hal pemodelan hubungan temporal yang panjang serta sering kali membutuhkan ekstraksi fitur manual yang tidak efisien. Selain itu, ML klasik juga rentan terhadap bias pemodelan. Hal tersebut mendorong dikembangkannya model ML menjadi model *Deep Learning* (DL).

Pendekatan DL muncul sebagai perluasan dari ML dengan struktur jaringan saraf berlapis yang memungkinkan pembelajaran fitur secara *end-to-end* (Goodfellow et al., 2016). Arsitektur jaringan DL yang dapat digunakan pada data sequential ialah *Recurrent Neural Network* (RNN). Arsitektur RNN dirancang secara khusus untuk data sekuensial seperti time
nemori internal untuk menyimpan informasi dari waktu ke waktu (Schäfer
). Varian lain dari RNN yakni Gated Recurrent Unit (GRU) yang
o, van Merriënboer, et al. (2014) dan Simple Recurrent Unit (SRU) oleh
terbukti efektif dalam memodelkan dinamika temporal pada data deret
ipun RNN memiliki kemampuan tersebut, struktur dasarnya tidak
engalami masalah *vanishing* dan *exploding gradient*, terutama pada
uk mengatasi masalah tersebut, Hochreiter & Schmidhuber (1997),

memperkenalkan pendekatan *Long Short-Term Memory* (LSTM) dengan struktur gate (input, forget, output). LSTM secara efektif mengontrol informasi yang disimpan, dilupakan, dan diproses, menjadikannya sangat cocok untuk prediksi harga saham jangka panjang maupun pendek.

Berbagai studi telah mengonfirmasi keunggulan LSTM dibandingkan pendekatan lainnya dalam prediksi harga saham. Pilla & Mekonen (2025) menunjukkan bahwa integrasi LSTM dengan fitur teknikal seperti moving average dan volume transaksi meningkatkan akurasi prediksi hingga lebih dari 92%. Selain itu, studi oleh Pirani et al. (2022) membuktikan bahwa kinerja LSTM dan GRU mengungguli ARIMA dalam prediksi data saham multivariabel. Tidak hanya itu, Chen et al., (2023) juga mengembangkan arsitektur GRU dengan integrasi data antar saham dalam sektor yang sama untuk meningkatkan representasi dan menurunkan risiko *overfitting*.

Tantangan prediksi saham tidak hanya terkait pada arsitektur model, tetapi juga pada strategi representasi data dan tuning parameter. Data saham pada dasarnya multivariabel, dinamis, dan heterogen. Untuk memaksimalkan pembelajaran, pendekatan stacking dari beberapa varian RNN dikenal sebagai Stacked RNN seperti LSTM, GRU, dan SRU telah mulai diterapkan dalam model-model prediktif (Althelaya et al., 2018; Tuan et al., 2022; Zeng et al., 2022). Ensambel semacam ini mampu menggabungkan kekuatan masing-masing arsitektur dan memperkuat stabilitas hasil prediksi. Dalam proses pembentukan model yang optimal, penyesuaian parameter seperti jumlah unit neuron, tingkat dropout, batch size, dan learning rate memegang peranan krusial.

Penyempurnaan proses tuning hyperparameter dalam model deep learning, berbagai teknik optimasi berbasis kecerdasan buatan telah digunakan, antara lain *Random Search*, *Bayesian Optimization* (BO), *Particle Swarm Optimization* (PSO), dan *Grey Wolf Optimizer* (GWO). *Random Search* mengeksplorasi ruang parameter secara acak dan terbukti lebih efisien daripada *Grid Search* dalam menemukan kombinasi optimal, terutama ketika hanya sebagian parameter yang mempengaruhi performa model secara signifikan (Bergstra & Bengio, 2012). Sementara itu, BO membangun model probabilistik melalui proses gausian untuk memetakan ruang parameter dan mengevaluasi lokasi yang paling menjanjikan dengan jumlah iterasi minimum (Shahriari et al., 2016). Teknik berbasis populasi seperti PSO dan GWO juga banyak digunakan karena kemampuannya dalam menyeimbangkan eksplorasi dan eksploitasi. PSO meniru perilaku sosial partikel dalam kawanan untuk mencari solusi terbaik (Kennedy & Eberhart, 1995), sedangkan GWO mengadopsi strategi hierarkis dan telah menunjukkan efektivitas tinggi dalam optimasi parameter model prediktif (Mirjalili et al., 2014). Integrasi metode-metode ini ke dalam arsitektur seperti LSTM secara empiris terbukti meningkatkan akurasi dan kestabilan model prediksi harga saham dibandingkan

anual (Rokhsatyazdi et al., 2020).

n juga menunjukkan bahwa bukan hanya struktur jaringan dan entukan kinerja prediksi, tetapi juga kualitas data input. Dalam fitur teknikal seperti *Relative Strength Index* (RSI), *Exponential* MA), *Bollinger Bands*, dan lain-lain, menjadi penting untuk ntasi informasi. Penelitian yang menggunakan fitur teknikal untuk odel DL (Sahib et al., 2024; Vargas et al., 2018). Selain itu, ide

menggabungkan informasi data dari sumber fitur utama dan melakukan segmentasi pola berdasarkan karakteristik temporal mulai diperkenalkan dalam pendekatan compositional learning. Studi terdahulu menggunakan strategi komposisional dalam analisis time series, misalnya algoritma compositional correlation untuk mengintegrasikan korelasi lokal (Dikbas, 2018), dan arsitektur modular yang segmentasi berdasarkan komponen temporal untuk transfer adaptasi antar domain (Sun et al., 2025; Vavilthota et al., 2024). Ini sejalan dengan pendekatan penelitian ini yang menggunakan segmentasi fitur saham (open, high, low, close, volume) melalui jalur paralel independen dan menggabungkannya secara komposisional ke dalam model RNN modular.

Pengukuran kebaikan model melalui proses evaluasi performa model dengan penggunaan metrik seperti RMSE, MAE, dan MAPE seringkali dianggap cukup. Namun untuk sistem prediksi yang kompleks dan berbasis ensambel, evaluasi performa perlu diperluas dengan metrik yang mempertimbangkan sensitivitas terhadap perubahan input, kestabilan terhadap data baru, dan kemampuan generalisasi (Jafar & Lee, 2023; Li et al., 2019; Oral & Altun, 2020). Hal ini penting agar model yang dikembangkan tidak hanya unggul pada data latih, tetapi juga tangguh dalam lingkungan pasar yang nyata dan berubah-ubah.

Memperhatikan keseluruhan dinamika tersebut, mulai dari kompleksitas data saham, keterbatasan metode klasik, keunggulan arsitektur RNN modern, pentingnya tuning parameter, hingga segmentasi pola dan evaluasi performa. Penelitian ini diarahkan untuk membangun pendekatan prediksi harga saham berbasis deep learning yang adaptif, akurat, dan andal. Penekanan pada integrasi arsitektur, komposisi fitur, dan penggunaan metrik evaluasi yang konfrehensip menjadi kontribusi penting dalam memperkuat fondasi ilmiah prediksi pasar berbasis data.

1.2 Rumusan Masalah

Data pasar saham yang dikoleksi secara berkala dalam bentuk time series bersifat kompleks, berskala besar, dan menunjukkan keterkaitan temporal yang erat. Ketika melibatkan banyak variabel dan sumber data, model prediksi menghadapi tantangan dalam menyusun input yang informatif serta merancang arsitektur jaringan yang dapat mengakomodasi dinamika pasar. Pendekatan berbasis deep learning seperti RNN dan variannya, dengan dukungan strategi optimasi dan pemrosesan data yang tepat, berpotensi memberikan hasil prediksi yang lebih akurat dan stabil.

Dengan mempertimbangkan kompleksitas tersebut, rumusan masalah dalam penelitian ini adalah:

a. Bagaimana efektifitas arsitektur *stacked* RNN berbasis LSTM dengan pendekatan input modular dan paralel pada data saham multivariabel?
b. Bagaimana membangun arsitektur komposisional *stacked* modular RNN (LSTM, [illegible]asis segmentasi input paralel dengan pengoptimalan metaheuristik [illegible]an prediksi time series multivariabel yang akurat, stabil?

1.3 Tujuan Penelitian

Tujuan utama dari penelitian ini adalah untuk mengembangkan model prediksi harga saham berbasis data time series multivariabel dengan pendekatan DL yang adaptif dan akurat. Untuk mencapai tujuan tersebut, penelitian ini secara khusus bertujuan untuk:

a. Menguji efektifitas arsitektur *Stacked* RNN berbasis LSTM dengan pendekatan input modular dan paralel pada data saham multivariable.
b. Membangun arsitektur komposisional *stacked* modular RNN (LSTM, GRU, SRU) berbasis segmentasi input paralel dengan pengoptimalan metaheuristik untuk menghasilkan prediksi time series multivariabel yang akurat dan stabil.

1.4 Kegunaan Penelitian

Penelitian ini diharapkan dapat memberikan kontribusi dalam dua ranah utama, yaitu secara teoritis dan praktis.

1.4.1 Kegunaan Teoritis

Secara teoritis, penelitian ini memberikan kontribusi terhadap pengembangan ilmu pengetahuan dalam bidang pembelajaran mesin dan deep learning, khususnya pada implementasi dan pengembangan arsitektur RNN untuk peramalan data time series. Penelitian ini memperkaya kajian tentang representasi input multivariabel yang relevan dalam prediksi harga saham, serta menyajikan pendekatan sistematis dalam membangun dan mengoptimalkan arsitektur model RNN, termasuk LSTM, GRU, dan SRU, melalui penerapan teknik optimasi seperti *Random Search* dan GWO. Selain itu, pendekatan berbasis segmentasi dan parallel memperluas ruang pengembangan dalam konstruksi model prediktif berbasis compositional learning, yang dapat menjadi acuan dalam studi-studi selanjutnya.

1.4.2 Kegunaan Praktis

Secara praktis, hasil dari penelitian ini dapat dimanfaatkan dalam pengembangan sistem prediksi harga saham yang lebih akurat dan adaptif, baik untuk kebutuhan akademik, industri keuangan, maupun pengambilan keputusan berbasis data (*data-driven decision making*). Pendekatan yang diusulkan dalam penelitian ini dapat membantu analis pasar dan pengembang sistem keuangan dalam membangun model prediksi yang mampu mengintegrasikan berbagai indikator teknikal dan informasi multivariabel secara lebih optimal. Dengan demikian, penelitian ini dapat mendorong implementasi teknologi kecerdasan buatan dalam pengambilan keputusan investasi dan pengelolaan risiko keuangan secara lebih presisi.

Penelitian

difokuskan pada pengembangan model prediksi harga saham ries multivariabel dengan menggunakan pendekatan *deep learning*, ari *Recurrent Neural Network* (RNN). Ruang lingkup penelitian an pada aspek-aspek berikut:

1.5.1 Jenis dan Sumber Data

Penelitian ini menggunakan data historis harga saham dalam format time series yang diperoleh dari sumber publik terpercaya seperti Yahoo Finance. Data yang digunakan mencakup variabel harga (*open, high, low, close, volume*), serta indikator teknikal hasil rekayasa fitur, seperti MA, EMA, RSI, MACD, Volume *Weighted Average Price* (VWAP), *High-Low*, *Moving Average Convergence Divergence* (MACD), *Average True Range* (ATR), *On Balance Volume* (OBV), *Bollinger Bands*, *Stochastic*-K dan *Stochastic*-D. Studi difokuskan pada saham-saham tertentu yang tergabung dalam indeks saham besar seperti Indeks Hang Seng dan saham Perusahaan IT seperti Google, Apple dan Microsoft.

1.5.2 Model dan Arsitektur yang Dikaji

Penelitian ini mengkaji dan membandingkan arsitektur RNN klasik dan variannya, termasuk LSTM, GRU, dan SRU. Selain itu, dilakukan pendekatan stacking internal model dan antar model serta eksplorasi pada komposisi model melalui mekanisme konkatenasi.

1.5.3 Teknik Optimasi Parameter

Untuk memperoleh konfigurasi model yang optimal, penelitian ini menggunakan metode tuning hyperparameter berbasis *Random Search* dan *Grey Wolf Optimizer* (GWO). Ruang lingkup tuning mencakup jumlah unit neuron, tingkat *dropout, learning rate, batch size*, dan *epoch*.

1.5.4 Cakupan Evaluasi Model

Evaluasi kinerja model dilakukan berdasarkan akurasi prediksi harga saham, yang diukur melalui metrik evaluasi seperti *Root Mean Square Error* (RMSE) dan *Mean Absolute Percentage Error* (MAPE), Koefisien Determinasi (R^2), *Root Mean Square Percentage Error* (RMSPE), *Nash-Sutcliffe Efficiency* (NSE), Percent Bias (PBIAS), *Willmott's Index of Agreement* (WI). Evaluasi dilakukan pada data uji yang dipisahkan dari data latih untuk mengukur kemampuan generalisasi model.

1.5.5 Batasan Analisis

Penelitian tidak membahas faktor-faktor fundamental seperti laporan keuangan, berita pasar, atau sentimen media sosial, dan tidak mencakup intervensi eksternal seperti krisis ekonomi atau kebijakan makro. Fokus penelitian dibatasi pada data historis kuantitatif dan indikator teknikal yang bersifat numerik.

tian

menawarkan kebaruan (*novelty*) yang signifikan dalam ranah
ham berbasis data time series multivariabel dengan pendekatan
n utamanya adalah merancang metode peramalan berbasis deep
erja tinggi melalui pengembangan arsitektur *Recurrent Neural*
ariannya dengan model komposisional *Stacked*-RNN (atau, model

RNN bertingkat terkomposisi) sebagai ***novelty* awal**. Untuk mencapai tujuan tersebut pertama-tama dirancang *Stacked*-RNN berbasis LSTM dengan segmentasi input parallel dari fitur utama dan fitur rekayasa sebagai ekperiman awal. Langkah selanjutnya adalah merancang arsitektur komposisional *stacked* modular RNN dengan menerapkan model LSTM, GRU, dan SRU dengan basis segmentasi input parallel yang sama. Arsitektur yang dirancang untuk mengatasi keterbatasan pendekatan-pendekatan sebelumnya yang umumnya hanya memanfaatkan satu jenis arsitektur. Selain itu, pendekatan sebelumnya hanya mencakup fitur input yang sempit (fitur utama dan fitur rekayasa), serta strategi optimisasi yang masih bersifat konvensional. Rancangan arsitektur tersebut dibangun dengan unsur pengembangan sebagai berikut.

a. Strategis Arsitektur RNN secara Bertingkat (*Stacked* RNN)

Penelitian ini mengembangkan dan mengeksplorasi kombinasi arsitektur *Recurrent Neural Network* (RNN), khususnya LSTM, GRU, dan SRU, dalam bentuk model bertingkat (*stacked*) yang dirancang secara modular dan komposisional, dimana setiap arsitektur dikonstruksi secara homogen. Pendekatan ini berbeda dari studi-studi sebelumnya yang umumnya terbatas pada penggunaan satu jalur input atau satu arsitektur tunggal tanpa mempertimbangkan efek struktural dari stacking modular dan segmentasi fitur secara paralel.

b. Pemanfaatan Komposisi dan pemisahan saluran informasi Saham Multi Fitur

Penelitian ini menerapkan pendekatan *compositional learning* berbasis segmentasi input dari pola data time series terhadap masing-masing fitur utama saham. Berbeda dari pendekatan konvensional yang cenderung menggunakan fitur secara agregat dalam satu jalur, penelitian ini mengusulkan strategi penggabungan pola *(composite pattern)* dengan memisahkan dan memproses masing-masing fitur—seperti *open, high, low, close*, dan *volume*—melalui jalur paralel yang independen. Strategi segmentasi modular ini masih jarang diterapkan dalam studi prediksi harga saham, yang umumnya terbatas pada model univariabel atau multivariabel tanpa struktur input yang terdistribusi.

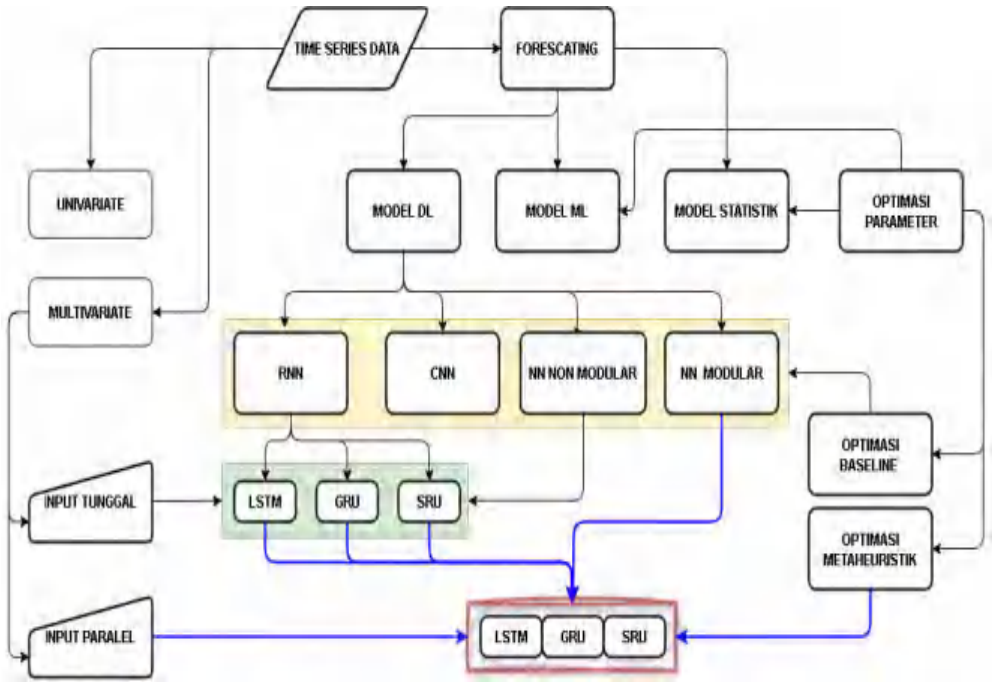
c. Integrasi Teknik Optimasi Modern dalam *Tuning Hyperparameter*

Penggunaan kombinasi teknik optimasi hyperparameter cerdas, yaitu *Random Search*, dan *Grey Wolf Optimizer* (GWO), yang secara komprehensif dibandingkan efektivitasnya dalam proses tuning model. Integrasi dan evaluasi perbandingan teknik optimasi ini dalam konteks peramalan harga saham berbasis *stacked*-RNN merupakan pendekatan yang relatif baru dan belum banyak diteliti secara sistematis.

d. Pendekatan Terpadu antara Representasi Data, Arsitektur Model, dan Strategi

holistik yang mengintegrasikan tiga komponen penting: ltivariabel berbasis fitur teknikal, desain arsitektur model bertingkat, s metrik akurasi seperti RMSE, MAPE, dan R-kuadrat, dll. Sinergi :an model prediksi yang lebih adaptif terhadap dinamika pasar dan generalisasi yang lebih kuat.

Secara keseluruhan, penelitian ini memberikan kerangka pengembangan sistem prediksi harga saham yang sistematis dan dapat direplikasi. Melalui penggabungan representasi fitur, pemilihan arsitektur yang fleksibel, serta teknik tuning dan evaluasi yang optimal, penelitian ini diharapkan menjadi pijakan ilmiah bagi studi lanjutan dalam bidang time series *forecasting* dan *intelligent financial systems*. Untuk melihat dengan jelas originalitas penelitian ini, berikut visualiasi kedudukan dari inovasi yang diusulkan.



Gambar 1.1 Bagan Originalitas Penelitian

1.7 Kajian Teori

1.7.1 Kajian Data Time Series Multivariabel

a. Pengertian dan karakteristik data time series

Data time series adalah sekumpulan pengamatan terhadap suatu variabel yang dikumpulkan dalam urutan waktu yang tetap dan teratur. Setiap nilai dalam deret waktu diasosiasikan dengan waktu $t \in \{1, 2, 3, \ldots, T\}$ dengan T adalah panjang deret (Hamilton, 1994). Representasi matematis dari data time series univariat dinyatakan

$$Z_t = \{z_1, z_2, z_3, \ldots, z_T\} \quad (1)$$

ilai observasi pada waktu ke-t. Dalam pendekatan statistik klasik, g umumnya dikaji dalam data time series meliputi:

Mean Function (fungsi rata-rata):

$$\mu_t = E(Z_t) \tag{2}$$

Fungsi variansi:

$$\gamma_{t,t} = \mathrm{Var} Z_t \tag{3}$$

Fungsi Autocovarian:

$$\gamma_{t,s} = \mathrm{Cov} Z_t, Z_s = E[(Z_t - \mu_t)(Z_s - \mu_s)] \tag{4}$$

Fungsi Autocorrelation:

$$\rho_{t,s} = \frac{\mathrm{Cov}(Z_t, Z_s)}{\sqrt{\mathrm{Var}(Z_t)\mathrm{Var}(Z_s)}} \tag{5}$$

Data time series umumnya meliputi harga historis (*open, high, low, close*), dan *volume* perdagangan dalam konteks pasar saham,, dan indikator teknikal seperti *moving average* (MA), *exponential moving average* (EMA), atau *relative strength index* (RSI). Karakteristik khas dari data time series saham adalah non-stasioner, bersifat musiman, dan sering menunjukkan volatilitas tinggi, sehingga tidak dapat secara optimal dianalisis menggunakan model linier sederhana seperti regresi biasa (Tsay, 2014).

Studi-studi terdahulu seringkali memodelkan harga saham hanya berdasarkan satu variabel tunggal (biasanya harga penutupan) dengan pendekatan univariat, tanpa memperhitungkan variabel lain yang memiliki hubungan dinamis. Selain itu, kebanyakan pendekatan konvensional mengabaikan struktur segmentasi waktu dan korelasi antar entitas saham, yang dapat mengandung informasi laten penting (Hewamalage et al., 2021; Shih et al., 2019).

Penelitian ini memberikan kontribusi dengan menyusun representasi input multivariabel yang mencakup variabel harga, volume, serta indikator teknikal yang relevan. Selain itu, digunakan pendekatan segmentasi dan komposisi data untuk membentuk pola historis dari fitur utama. Dengan demikian, model prediksi yang dikembangkan dapat memanfaatkan struktur informasi yang lebih dalam dari sekadar urutan nilai harga, sehingga diharapkan meningkatkan akurasi dan stabilitas prediksi dalam konteks pasar yang dinamis.

b. Perbedaan univariat dan multivariat dalam konteks finansial

Pendekatan univariat dalam analisis time series mengandalkan satu variabel utama sebagai dasar peramalan, umumnya harga penutupan (*close price*). Secara matematis, model univariat hanya menggunakan informasi masa lalu dari variabel itu sendiri, misalnya:

$$\widehat{z_{t+1}} = f\left(z_t, z_{t-1}, \dots, z_{t-p}\right) \tag{6}$$

nilai prediksi harga saham di masa depan berdasarkan nilai-nilai
abel tunggal z, dan p adalah lag order. Pendekatan ini sederhana
ı karena struktur datanya minimal, tetapi mengabaikan informasi

penting dari variabel lain yang bisa mempengaruhi pergerakan harga, seperti volume, volatilitas, atau indikator teknikal (Tsay, 2014).

Pendekatan multivariat memanfaatkan lebih dari satu variabel input yang saling berkorelasi secara temporal. Dalam konteks prediksi harga saham, ini mencakup kombinasi harga *open, high, low, close, volume*, dan fitur-fitur teknikal yang direkayasa. Model multivariat dapat diformulasikan sebagai:

$$\widehat{z_{t+1}} = f\left(x_t, x_{t-1}, \dots, x_{t-p}\right) \tag{7}$$

dengan $x_t = \left[x_t^{(1)}, x_t^{(2)}, \dots, x_t^{(n)}\right]$ sebagai vektor input multivariabel pada waktu t, yang terdiri dari n fitur. Pendekatan ini memungkinkan model menangkap hubungan lintas fitur dan mengidentifikasi pola ko-evolusi antar variabel dalam horizon waktu tertentu (Lütkepohl, 2005).

Model multivariat telah terbukti memberikan peningkatan performa signifikan dalam prediksi pasar saham dibandingkan model univariat. Studi oleh Cirstea et al. (2018) menunjukkan bahwa penggunaan input multivariabel dengan jaringan DL secara konsisten meningkatkan akurasi prediksi dibandingkan pendekatan univariat. Selain itu, pendekatan multivariat juga memungkinkan integrasi informasi sektoral atau lintas saham yang saling memengaruhi, yang seringkali tidak terjangkau oleh model linier tradisional (Yin & Barucca, 2021).

Namun demikian, tantangan dalam pendekatan multivariat meliputi kompleksitas dimensi fitur, potensi *overfitting*, serta perlunya teknik normalisasi dan segmentasi yang tepat. Penelitian ini menjawab tantangan tersebut dengan membangun representasi input multivariabel yang terstruktur, menggabungkan indikator teknikal, serta menyusun pola komposisi dari beberapa saham yang berkorelasi. Dengan pendekatan ini, model prediksi diharapkan dapat lebih adaptif terhadap dinamika pasar dan menghasilkan estimasi yang lebih akurat dan stabil dibandingkan pendekatan univariat konvensional.

c. Representasi dan Transformasi Fitur Teknikal

Prediksi harga saham berbasis data time series, representasi input memiliki peran yang sangat penting dalam menentukan kemampuan model untuk menangkap pola pergerakan harga yang kompleks dan dinamis. Salah satu strategi yang umum dilakukan dalam feature engineering adalah menambahkan indikator teknikal sebagai fitur tambahan di luar harga saham dasar (*open, high, low, close*) dan *volume*. Indikator teknikal ini dirancang untuk mengekstraksi informasi seperti tren harga, momentum, volatilitas, dan tekanan beli/jual berdasarkan data historis (Murphy, 1999).

Beberapa indikator teknikal populer yang digunakan dalam penelitian ini meliputi:

A) (Murphy, 1999), yaitu rata-rata harga penutupan dalam periode

$$MA_t^{(n)} = \frac{1}{n}\sum_{i=0}^{n-1} P_{t-i}\,, \tag{8}$$

pada waktu $\boldsymbol{t}$,

Exponential Moving Average (EMA) (Murphy, 1999), memberikan bobot lebih besar pada data terbaru:

$$EMA_t = \alpha \cdot P_t + (1 - \alpha) \cdot \text{EMA}_{t-1},\ \alpha = \frac{2}{n+1} \tag{9}$$

Relative Strength Index (RSI) (Wilder Jr., 1978), mengukur kekuatan tren naik terhadap tren turun.

$$RSI_t = 100 - \left(\frac{100}{1+RS_t}\right),\ RS_t = \frac{\text{Average Gain}}{\text{Average Loss}} \tag{10}$$

Moving Average Convergence Divergence (MACD) (Gerald, 2005), membandingkan dua EMA.

$$MACD_t = \text{EMA}_{12,t} - \text{EMA}_{26,t},\ \text{Signal}_t = \text{EMA}_9(\text{MACD}_t) \tag{11}$$

Volume Weighted Average Price (VWAP) (Madhavan, 2002), harga rata-rata berdasarkan volume.

$$VWAP_t = \frac{\sum_{i=1}^{t}(P_i \cdot V_i)}{\sum_{i=1}^{t} V_i} \tag{12}$$

P_i: Harga rata-rata $\frac{\text{High+Low+Close}}{3}$

V_i: Volume pada waktu i.

High–Low Range, selisih antara harga tertinggi dan terendah harian:

$$Range_t = \text{High}_t - \text{Low}_t \tag{13}$$

Average True Range (ATR) (Wilder Jr., 1978), ukuran volatilitas.

$$ATR_t = \frac{1}{n}\sum_{i=0}^{n-1} \text{TR}_{t-i} \tag{14}$$

dimana

$$TR_t = \max\left(\text{High}_t - \text{Low}_t, \left|\text{High}_t - \text{Close}_{t-1}\right|, \left|\text{Low}_t - \text{Close}_{t-1}\right|\right)$$

On Balance Volume (OBV) (Granville, 2018), akumulasi volume berdasarkan arah harga.

$$OBV_t = \begin{cases} OBV_{t-1} + V_t, & \text{jika } P_t > P_{t-1} \\ OBV_{t-1} - V_t, & \text{jika } P_t < P_{t-1} \\ OBV_{t-1}, & \text{jika } P_t = P_{t-1} \end{cases} \tag{15}$$

Bollinger Bands (Bollinger, 2001), batas harga atas dan bawah dari moving average.

$$Upper_t = \text{MA}_t + k \cdot \sigma_t,\ \text{Lower}_t = \text{MA}_t - k \cdot \sigma_t \tag{16}$$

dari harga penutupan dalam periode n,

ɔr (%K dan %D) (Murphy, 1999), mengukur posisi harga relatif

$$\%K_t = \frac{P_t - \text{Low}_n}{\text{High}_n - \text{Low}_n} \times 100,\ \%D_t = \text{MA}_3(\%K) \quad (17)$$

Penggunaan indikator-indikator ini memungkinkan model untuk mengenali pola tren, tekanan pasar, dan anomali pergerakan harga secara lebih kaya dibandingkan hanya dengan input harga dasar. Meskipun telah banyak digunakan dalam riset sebelumnya, pendekatan manual dalam pemilihan indikator sering kali tidak mempertimbangkan sensitivitas terhadap hasil prediksi dan dapat meningkatkan kompleksitas model tanpa kontribusi signifikan terhadap performa (Patel et al., 2015). Penelitian ini mengatasi hal tersebut dengan membangun struktur input yang sistematis, mengintegrasikan indikator-indikator tersebut secara terukur, serta menguji pengaruh masing-masing indikator.

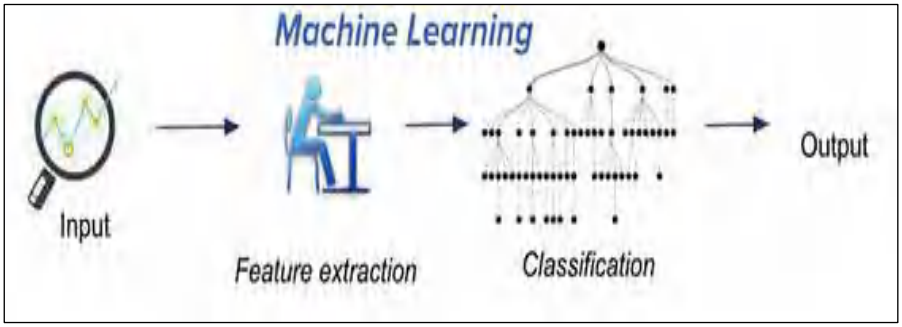
1.7 2 Deep Learning

a. Deep Neural Network

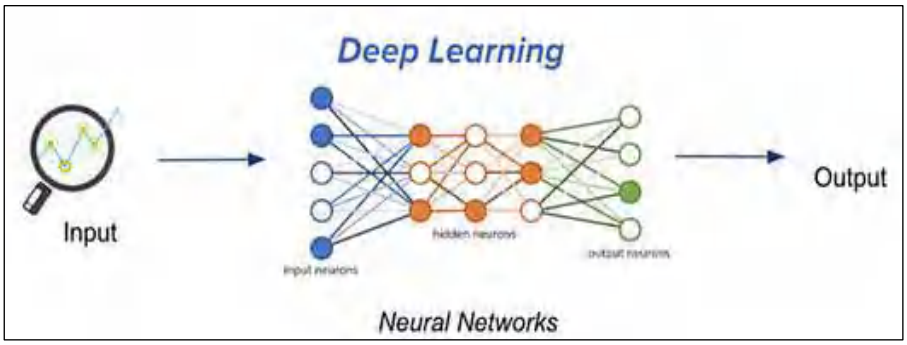
Kecerdasaan Buatan (*Artificial Intelligence (AI*)) merupakan revolusi computer yang memberikan perubahan secara quantum terhadap dunia terknologi dalam dekade ini, diantara AI yang telah berkembang adalah *Machine Learning* (ML). ML merupakan metode yang memiliki kemampuan dengan secara otamatis dapat melakukan pembelajaran dan meningkatkan pengalaman tanpa diprogram secara eksplisit, kemampuan ini dapat mengoptimalkan performa system berdasarkan hasil pembelajaran dan pengalaman yang ditemukan dari data sampel atau data histori. ML berfokus pada pengembangan program komputasi yang dapat mengakses data, kemudian terbentuk menjadi rangkaian teknis yang direpresentasikan dalam suatu algoritma pembelajaran sehingga mampu untuk menangani dan memberikan hasil prediksi terhadap data yang terkumpul dalam jumlah besar atau *data mining* (Alpaydin, 2009).

Perkembangan dari ML, ketika kemampua pembelajaran yang didasarkan pada kedalaman struktur dari jaringan syaraf tiruan dengan perbelajaran terhadap hirarki konsep dengan jumlah lapisan yang banyak dikenal sebagai *Deep Learning* (DL) (Goodfellow et al., 2016). Penambahan lebih banyak lapisan pada teknik *deep learning* memberikan arsitektur yang sangat kuat untuk *supervised learning*, model pembelajaran tersebut bisa mewakili data berlabel dengan lebih baik.

Model pembelajaran dari algoritma ML melalui konsep jaringan syaraf tiruan dengan melibatkan jaringan tersembunyi yang lebih banyak berdampak pada peningkatan dari kemampuan mesin (komputer) untuk mampu belajar dengan kecepatan, akurasi dan skala data yang lebih besar. Prinsip ini terus berkembang pada komunitas riset dan industri sebagai alat pemecahan masalah terkait dengan karakteristik dan variabilitas data dengan ukuran besar, sehingga DL sangat banyak elitian terkait dengan *computer vision, speech recognition, dan :essing.*



Gambar 1.2 Proses klasifikasi pada *Machine Learning*



Gambar 1.3 Tipikal proses pada *Deep Learning*

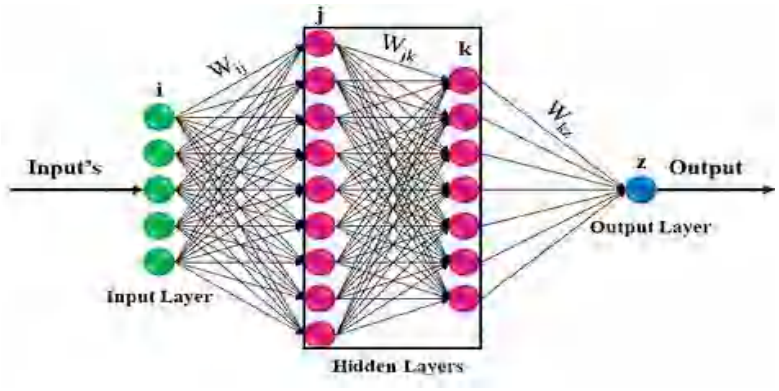
Gambar 1.2 dan Gambar 1.3 mengilustrasikan perbedaan proses yang terjadi pada algoritma *ML* dan *DL*. Sebagai ilustrasi, pendekatan ML melakukan ekstraksi fitur secara manual yang dapat mengklasifikasikan suatu objek menjadi target 1 dan 0, proses ini menunjukkan algoritma yang terjadi pada ML. Berbeda dengan proses algoritma yang terjadi pada DL, ekstraksi fitur dilakukan secara otomatis atau dengan kata lain DL tidak selalu membutuhkan data terstruktur, karena DL mampu mempelajari representasi hirarki dari data itu sendiri dan data yang banyak dapat diskalakan dalam dimensi tertentu.

Deep Learning berkembang dengan memaksimalkan karakteristik dari jaringan syaraf tiruan melalui model graf berlapis dengan neuron yang banyak disebut *Multi-Layer Perceptron* (MLP). MLP sebagai pengembangan dari perceptron tunggal dengan generate nilai bobot yang lebih baik dari pemodelan lainnya, sehingga memberikan kinerja terhadap hasil klasifikasi yang lebih akurat. Selain karakteristiknya yang memiliki layer persentron multi, juga memiliki beberapa lapisan ataupun *hidden layer* yang terletak diantara ruang *input* dan *output layer*.

ng dimiliki MLP terdiri dari tiga unit bagian yakni: lapisan masukan
an tersembunyi (*hidden layer*), lapisan luaran (*output layer*) (Simon

menerima data masukan merupakan lapisan masukan (*input layer*)
uron yang menghubungkan dengan variabel *x* atau variabel input.
on pada struktur ini terhubung dengan *neuron* pada lapisan

2) Bagian yang melakukan proses pembelajaran atau mengenali karakateristik data terjadi pada lapisan tersembunyi (*hidden layer*) dimana bagian ini terdiri dari neuron yang menerima data dari lapisan masukan.
3) Hasil kalkulasi dari variable input *x* menjadi nilai target *z* terproses pada lapisan luaran (*output layer*), bagian ini terdiri dari *neuron* yang menerima data dari lapisan tersembunyi.



Gambar 1.4 Arsitektur Multilayer Perceptron

Gambar 1.4 menampilkan visualisasi dari MLP dengan keterangan: vektor input ke-*i* yang masuk pada input layer. W_{ij} adalah matriks bobot setiap neuron yang terhubung ke *hidden layers*. W_{jk} adalah matriks bobot pada setiap neuron pada lapisan dari *hidden layers* pertama ke berikutnya. W_{kz} adalah matriks nilai aktivasi. *z* adalah nilai output hasil perhitungan dari input oleh suatu fungsi aktivasi.

b. Perkembangan *machine learning* dan *deep learning* dalam finansial

Dalam satu dekade terakhir, pendekatan *machine learning* (ML) dan *deep learning* (DL) telah menjadi paradigma baru dalam bidang keuangan, khususnya dalam prediksi harga saham dan analisis risiko pasar. *Machine learning* menawarkan fleksibilitas dalam mengenali pola nonlinier dan mengatasi keterbatasan asumsi stasioneritas serta linieritas yang mendasari metode statistik konvensional. Teknik seperti *support vector machines* (SVM), *decision trees*, dan *random forests* telah banyak digunakan untuk klasifikasi arah harga saham, identifikasi sinyal beli/jual, serta deteksi anomali pasar (Atsalakis & Valavanis, 2009; Patel et al., 2015a). ML mampu mengolah data pasar yang besar, cepat, dan tidak terstruktur, dengan pendekatan berbasis data-driven daripada model-driven.

Seiring dengan meningkatnya ketersediaan data finansial berfrekuensi tinggi dan kemampuan komputasi, pendekatan DL mulai menggantikan ML klasik dalam uangan. Arsitektur seperti *artificial neural networks* (ANN), *networks* (CNN), dan terutama *recurrent neural networks* (RNN) M, GRU) telah terbukti lebih unggul dalam menangkap hubungan nsi jangka panjang, dan struktur nonlinier pada data time series 17; Fischer & Krauss, 2018a). DL mampu melakukan ekstraksi fitur ntah tanpa memerlukan rekayasa fitur manual secara eksplisit. Ini

memberikan keunggulan dalam menyusun representasi kompleks dari data historis saham, termasuk pola tren, volatilitas, dan dinamika korelasi lintas saham.

Selain itu, integrasi model deep learning dengan teknik optimasi, ensemble learning, dan transfer learning telah menunjukkan performa prediksi yang lebih stabil, adaptif, dan akurat pada data pasar yang sangat fluktuatif. Penelitian-penelitian terkini juga mulai memanfaatkan arsitektur hybrid dan komposisi model (misalnya *stacked* LSTM-GRU, CNN-LSTM, atau attention-based RNN) yang dirancang khusus untuk meniru mekanisme kognitif pengambilan keputusan dalam pasar finansial (Jebaraj et al., 2024; H. Song & Choi, 2023; Tripathy et al., 2023). Pendekatan ini tidak hanya memperbaiki akurasi, tetapi juga memperluas kemampuan model dalam menyerap informasi dari berbagai dimensi data dari harga, volume, indikator teknikal, serta sentimen pasar.

Untuk memperjelas evolusi pendekatan prediksi saham dari metode statistik ke pembelajaran mendalam, Tabel.1 berikut merangkum karakteristik, kekuatan, dan keterbatasan masing-masing pendekatan beserta referensinya:

Tabel 1.1 Perbandingan Pendekatan Prediksi Harga Saham: Statistik, Machine Learning, dan Deep Learning

Pendekatan	Metode	Kapabilitas	Kelebihan	Kelemahan	Referensi
Statistik	ARIMA, GARCH, Prophet	Linier, volatilitas musiman	Sederhana, interpretatif	Tidak tangani nonlinieritas, korelasi kompleks	Kaninde et al. (2022); Zíková & Veselá (2023)
ML	SVM, RF, XGBoost	Nonlinier, supervised	Adaptif, cocok data tabular	Butuh rekayasa fitur, tidak eksplisit tangani time-dependence	Brunda & Khan (2020); Shah & Thaker (2024); Xu et al. (2022)
DL	LSTM, GRU, CNN, Transformer	Multivariabe, sekuensial, deep features	Tangani long-term dependencies, fitur otomatis	Butuh data besar, rentan overfitting	Fu et al. (2023; Seabe et al. (2023); H. Song & Choi (2023)

c. Studi-studi prediksi saham berbasis deep learning

Pendekatan deep learning telah menunjukkan potensi besar dalam prediksi harga saham karena kemampuannya menangani karakteristik data yang bersifat [illegible] serta mengandung ketergantungan temporal jangka panjang. Salah [illegible]paling banyak digunakan dalam ranah ini adalah *Long Short-Term* [illegible]ng dirancang untuk mengatasi permasalahan vanishing gradient [illegible] pada jaringan saraf rekuren klasik. Fischer & Krauss (2018) [illegible] LSTM secara signifikan mengungguli metode konvensional seperti [illegible]*andom forest* dalam klasifikasi arah indeks harga S&P500, dengan

tingkat akurasi mencapai lebih dari 60% dalam horizon harian, angka yang tergolong signifikan dalam pasar keuangan yang bersifat dinamis.

Sejumlah studi telah mengimplementasikan LSTM dalam berbagai pendekatan praktis. Rony et al. (2019) menerapkan LSTM untuk memprediksi harga saham di Dhaka Stock Exchange (DSE) dengan menggunakan data hasil ekstraksi web scraping, serta menggabungkan data sentimen berita dari situs seperti DSE dan Lonkabangla dengan algoritma *Naive Bayes* untuk mengurangi *noise* dan mengklasifikasi sentimen. Kapoor et al. (2020) menggunakan arsitektur LSTM untuk memprediksi harga saham Apple pada rentang waktu 2014–2019. Evaluasi menggunakan metrik MSE menunjukkan hasil sekitar 1%, yang meskipun tidak terlalu akurat, tetap memberikan kontribusi sebagai pendekatan prediksi jangka pendek harian.

Pengembangan arsitektur berbasis pola juga telah dilakukan oleh D. Song et al., (2020) melalui integrasi LSTM dengan algoritma pembelajaran tak terawasi (*unsupervised learning*) untuk prediksi indeks saham Korea. Hasil evaluasi menunjukkan performa model lebih baik dibandingkan dengan vanilla RNN dan LSTM konvensional. Selain itu H. K. Choi & Student (2018) mengintegrasikan metode ARIMA dengan LSTM dan menunjukkan adanya peningkatan performa prediksi melalui nilai koefisien korelasi yang lebih tinggi.

Aspek optimasi hiperparameter turut mendapat perhatian penting. (Rokhsatyazdi et al., 2020) mengusulkan pengoptimalan sepuluh parameter utama dalam jaringan LSTM—meliputi ukuran jendela waktu, *batch size*, jumlah unit LSTM pada *hidden layer*, jumlah layer tersembunyi (baik LSTM maupun *Dense*), *dropout*, serta algoritma pelatihan. Penggunaan algoritma *Differential Evolution* (DE) berhasil menurunkan nilai RMSE hingga 8.092, melampaui model statistik seperti NAIVE, ETS, dan SARIMA.

Integrasi data eksternal seperti sentimen pasar juga terbukti meningkatkan performa model. Jabeen et al. (2020) menggunakan data Twitter terkait sentimen pasar selama pandemi COVID-19 dan menunjukkan bahwa integrasi dengan LSTM dapat meningkatkan akurasi prediksi berdasarkan evaluasi MAE, MSE, dan RMSE. Perbandingan algoritmik dilakukan oleh Kumar et al. (2021) yang menunjukkan bahwa performa LSTM lebih baik dibandingkan algoritma *Prophet* dalam memprediksi harga saham berdasarkan data historis enam tahun. Penelitian oleh Mohsin Kabir et al. (2022) mengimplementasikan pembaruan pada update gate dalam arsitektur LSTM untuk prediksi saham di pasar Dhaka, dan hasil evaluasi menunjukkan penurunan signifikan pada MAE, RMSE, dan MAPE. Cao et al. (2019) mengusulkan dua pendekatan hybrid berbasis *Empirical Mode Decomposition* (EMD) dan *Complete Ensemble Empirical Mode Decomposition with Adaptive Noise* (CEEMDAN) yang dikombinasikan dengan LSTM. Studi tersebut menunjukkan bahwa model hybrid melampaui performa LSTM tunggal, *Support Vector Machine* (SVM), dan *Multi-Layer Perceptron* (MLP) dalam konteks

e & Park (2022) melakukan eksperimen dengan dua jaringan LSTM
ng waktu (T) untuk memprediksi satu langkah ke depan pada tiga
currency, dan komoditas. Jaringan LSTM pertama digunakan untuk
poch terbaik, sementara jaringan kedua dilatih ulang untuk prediksi.
kan dengan metode lain seperti Extended Kalman Filter,

autoregressive model, dan ARIMA. Dalam studi serupa, N. Singh et al., (2022) mengusulkan model hybrid berbasis LSTM dan menunjukkan hasil yang sebanding dalam prediksi harga saham. Selain LSTM, *Gated Recurrent Uni*t (GRU) juga semakin banyak digunakan karena arsitekturnya yang lebih sederhana dan efisiensi komputasi yang lebih baik. Pang (2024) menunjukkan bahwa GRU memberikan hasil prediksi jangka pendek yang setara atau lebih unggul dari LSTM pada saham Apple. Pendekatan lainnya adalah dengan menggunakan *Convolutional Neural Network* (CNN), seperti yang dilakukan oleh Sezer et al. (2020), yang mengubah data OHLC menjadi citra dan kemudian memprosesnya dengan CNN untuk menghasilkan sinyal perdagangan.

Gabungan arsitektur CNN–LSTM dan stacked RNN juga banyak diteliti karena kemampuannya menggabungkan kekuatan CNN dalam ekstraksi fitur dan LSTM dalam pemodelan sekuensial. Bao et al. (2017) menggunakan stacked autoencoder sebelum LSTM dan berhasil meningkatkan akurasi prediksi pada pasar saham Tiongkok. Studi terbaru oleh Mahdi et al. (2025) mengeksplorasi arsitektur Attention dan Transformer untuk mengatasi ketergantungan jangka panjang pada data deret waktu harga *Cryptocurrency*, dan hasilnya menunjukkan stabilitas dan interpretabilitas yang lebih baik dalam performa prediksi.

Sebagian besar studi tersebut masih menghadapi beberapa keterbatasan. Banyak pendekatan hanya menggunakan data univariat, tanpa mempertimbangkan hubungan antar variabel yang kompleks. Selain itu, proses optimasi hyperparameter kerap dilakukan secara manual atau hanya parsial, dan segmentasi data historis sering kali dilakukan tanpa eksplorasi fitur komposisi yang lebih dalam seperti indikator teknikal, sentimen publik, atau informasi fundamental perusahaan.

Berdasarkan kelemahan tersebut, terdapat ruang kontribusi luas untuk mengembangkan pendekatan prediksi harga saham berbasis stacked dan ensemble RNN, integrasi rekayasa fitur teknikal dan eksternal, serta penerapan algoritma tuning metaheuristik seperti PSO atau GWO guna meningkatkan akurasi dan generalisasi model prediktif lintas sektor. Pengembangan semacam ini diharapkan dapat meningkatkan kualitas pengambilan keputusan dalam pasar saham yang semakin kompleks dan volatil.

1.7.3 Fungsi Aktivasi dan Optimasi

a. Fungsi Aktivasi Sigmoid

Fungsi sigmoid didefinisikan sebagai:

$$\sigma(z) = \frac{1}{1+e^{-z}} \tag{18}$$

Fungsi ini memetakan input ke rentang (0,1), cocok untuk model yang membutuhkan interpretasi probabilitas, seperti gating mechanism dalam LSTM dan GRU

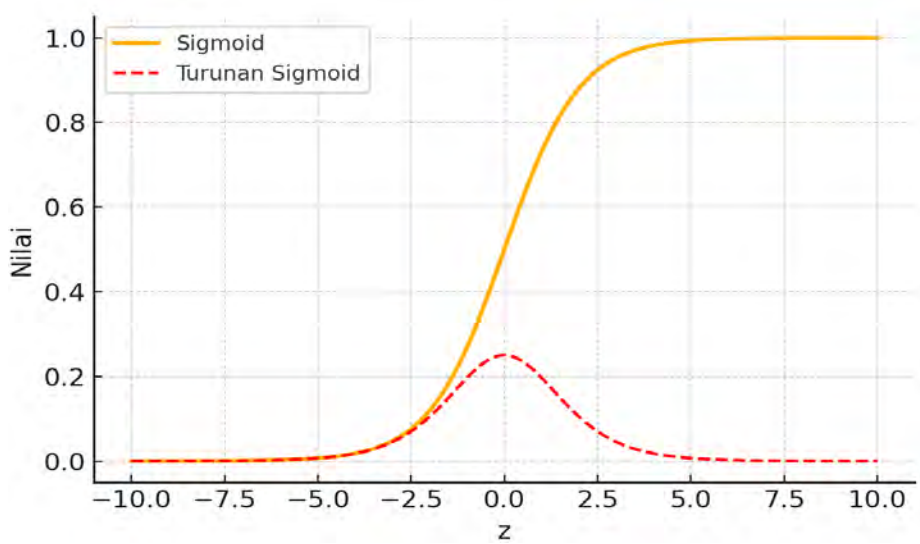
16).

fungsi sigmoid adalah:

$$\frac{\mathrm{d}}{dz}\sigma(z) = \sigma(z)\big(1-\sigma(z)\big) \tag{19}$$

id digunakan untuk mengubah input linear menjadi output nonlinier . Nilai sigmoid dapat ditafsirkan sebagai tingkat aktivasi atau

probabilitas, menentukan seberapa besar informasi yang akan diteruskan. Sebagaimana diperlihatkan pada Gambar 1.5, kurva sigmoid meningkat secara halus dan monoton, sementara turunannya memuncak di sekitar $z = 0$ dan mendekati nol di kedua ujung ekstrem. Pola ini menggambarkan potensi vanishing gradient, yang menjadi tantangan saat bobot besar menyebabkan output jenuh dan turunan menjadi sangat kecil.



Gambar 1.5 Kurva Fungsi Aktivasi Sigmoid dan Turunannya

b. Fungsi Aktivasi tanh

Fungsi tanh (hyperbolic tangent) adalah:

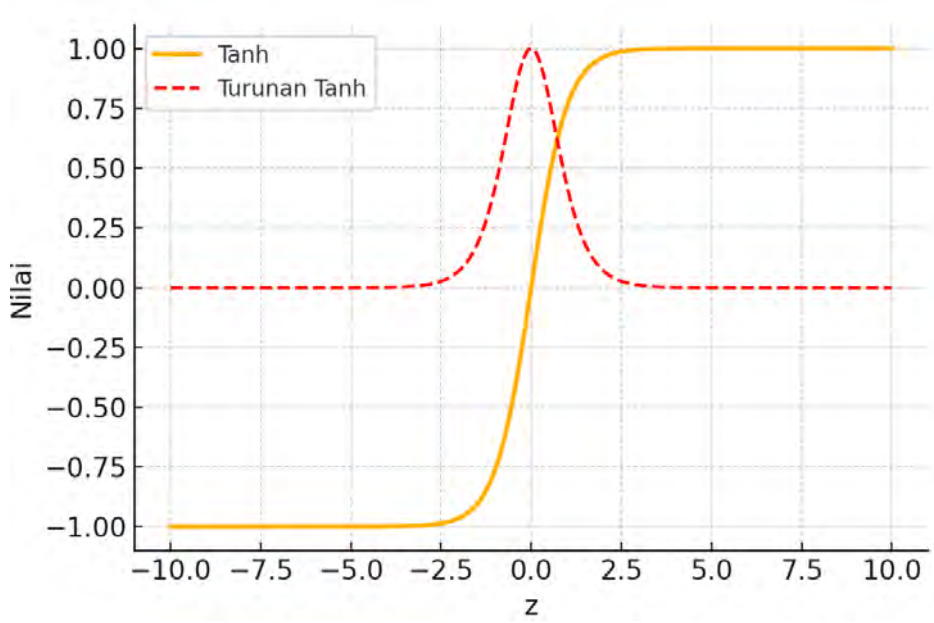
$$\tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} \tag{20}$$

Output-nya berada pada rentang (-1, 1), sehingga bersifat *zero-centered* dan lebih stabil dalam beberapa jaringan RNN (Lecun et al., 2015).

Turunan dari *tanh* digunakan untuk menghitung sensitivitas terhadap perubahan kecil dalam input:

$$\frac{\mathrm{d}}{dz}\tanh(z) = 1 - \tanh^{2}(z) \tag{21}$$

Fungsi aktivasi *tanh* menghasilkan output dalam rentang (−1,1), memberikan keuntungan karena lebih terpusat di nol, sehingga dapat mempercepat konvergensi selama pelatihan dan menjaga stabilitas gradien. Pada Gambar 1.6, terlihat bahwa kurva
ɔ sumbu vertikal, dengan nilai output yang mendekati -1 atau 1 pada
ırunannya, yang juga ditampilkan dalam gambar, mencapai nilai
(sekitar $z = 0$) dan menurun tajam di luar rentang tersebut,
itas terhadap input rendah pada nilai ekstrem, fenomena yang juga
ınishing gradient dalam pelatihan jaringan mendalam.



Gambar 1.6 Kurva Fungsi Aktivasi tanh dan Turunannya

c. Fungsi Optimasi ADAM

Dalam pelatihan model jaringan saraf, proses pembaruan parameter secara efisien dan stabil menjadi komponen penting untuk mencapai konvergensi yang optimal. Salah satu algoritma optimasi yang telah terbukti efektif dalam berbagai arsitektur jaringan, termasuk RNN, LSTM, dan GRU adalah ADAM (*Adaptive Moment Estimation*). ADAM diperkenalkan oleh Kingma & Ba (2014) sebagai metode stokastik berbasis gradien yang menggabungkan keunggulan dari dua algoritma sebelumnya, yaitu momentum dan RMSProp, dalam satu kerangka adaptif.

Secara formal, algoritma ADAM dapat dijelaskan melalui tahapan berikut: pertama, gradien dari fungsi loss terhadap parameter θ_t dihitung sebagai $g_t = \nabla_\theta L(\theta_t)$. Kemudian, estimasi momentum diperbarui menggunakan rumus $m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$, dan estimasi variansi diperoleh melalui $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$. Setelah dilakukan koreksi bias, yaitu $\widehat{m_t} = \frac{m_t}{1-\beta_1^t}$ dan $\widehat{v_t} = \frac{v_t}{1-\beta_2^t}$, parameter model diperbarui dengan persamaan:

$$\theta_{t+1} = \theta_t - \alpha \cdot \frac{\widehat{m_t}}{\sqrt{\widehat{v_t}}+\varepsilon} \tag{22}$$

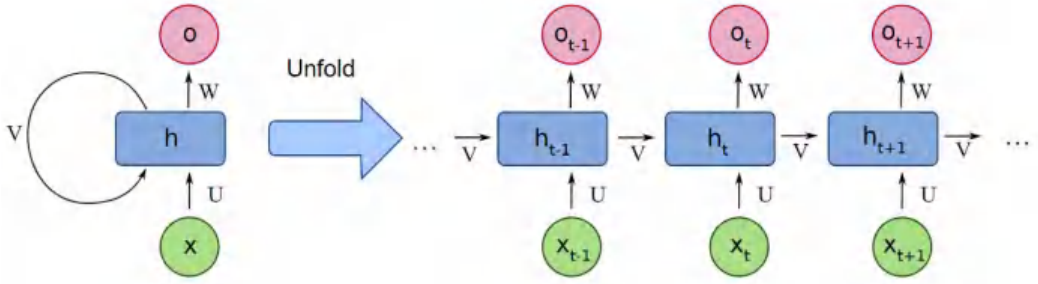
Keunggulan utama ADAM adalah kemampuannya menyesuaikan *learning rate* secara individual untuk setiap parameter, sehingga lebih tangguh terhadap noise dan Selain itu, parameter default yang direkomendasikan oleh Kingma .001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, dan $\varepsilon = 10^{-8}$, secara umum telah ng kompetitif tanpa memerlukan tuning tambahan yang kompleks. M menjadi algoritma pilihan dalam pelatihan model *deep learning* alam konteks sekuensial time series seperti prediksi harga saham, dalam menangani gradien yang fluktuatif antar time-step.

1.7.4 Arsitektur Recurrent Neural Network (RNN) dan Variannya

a. Konsep dasar RNN

Recurrent Neural Network (RNN) merupakan salah satu arsitektur dasar dalam *deep learning* yang dirancang khusus untuk menangani data berurutan (*sequential data*), seperti teks, sinyal, dan deret waktu (time series). Tidak seperti jaringan *feedforward*, RNN memiliki loop internal yang memungkinkan informasi dari waktu sebelumnya ditransmisikan ke waktu saat ini. Hal ini memungkinkan jaringan untuk "mengingat" konteks masa lalu dalam memproses data saat ini (Elman, 1990). Dengan kemampuan ini, RNN sangat cocok untuk pemodelan hubungan temporal seperti pada pergerakan harga saham, di mana data historis memengaruhi nilai saat ini maupun prediksi masa depan. RNN dirancang sebagai jaringan berulang yang memungkinkan nilai neuron dari hidden layer sebelumnya dimanfaatkan kembali sebagai input pada waktu berikutnya (Mikolov et al., 2014). Hal ini membuatnya banyak digunakan dalam berbagai domain, seperti pemrosesan bahasa alami (*natural language processing*), pengenalan suara, sintesis musik, hingga prediksi data finansial (K. Choi et al., 2017).

Struktur matematisnya RNN dibangun berdasarkan prinsip dari *feedforward neural network* (FNN), namun dengan tambahan mekanisme memori yang menyimpan hasil keluaran sebelumnya untuk mempengaruhi proses inferensi pada waktu berikutnya. Misalnya, pada suatu sekuens data input $X = (x_1, x_2, \dots, x_T)$, probabilitas dari input saat ini x_t tidak hanya bergantung pada input langsung, tetapi juga pada urutan sebelumnya, sehingga dapat dimodelkan sebagai $P(x_t|x_1, x_2, \dots, x_{t-1})$.



Gambar 1.7 Arsitektur RNN yang Di-unfold dalam Dimensi Waktu
Sumber: Adaptasi dari (Elman, 1990; Mikolov et al., 2014))

Berdasarkan Gambar 1.7 menunjukkan proses **unfolding** pada arsitektur RNN, di mana setiap langkah waktu t mengolah input x_t, menghasilkan *hidden state* h_t, dan memproduksi output o_t. Bobot U, V, dan W masing-masing digunakan untuk koneksi input ke *hidden layer*, *hidden-to-hidden* (recurrent), dan *hidden to output*. Proses ini memungkinkan RNN untuk "mengingat" informasi dari langkah sebelumnya melalui rulang. Tipe dari RNN terdiri dari *one to one* yaitu single input dan upakan tipe RNN yang sederhana atau terkadang disebut vanila a adalah single input dan multiple output atau disebut *one to many*. miliki mult input dan single ouput atau *many to one*. Terakhir *to many*, dimana inputnya multi dan outputnya pun multi.

Secara matematis, output tersembunyi pada waktu t, yaitu $\boldsymbol{H}_t$ dihitung berdasarkan input saat ini X_t dan hidden state sebelumnya $\boldsymbol{H}_{t-1}$ dengan:

$$\boldsymbol{H}_t = (\boldsymbol{W}_h \cdot [\boldsymbol{X}_t,\ \boldsymbol{H}_{t-1}] + \boldsymbol{b}_h) \tag{23}$$

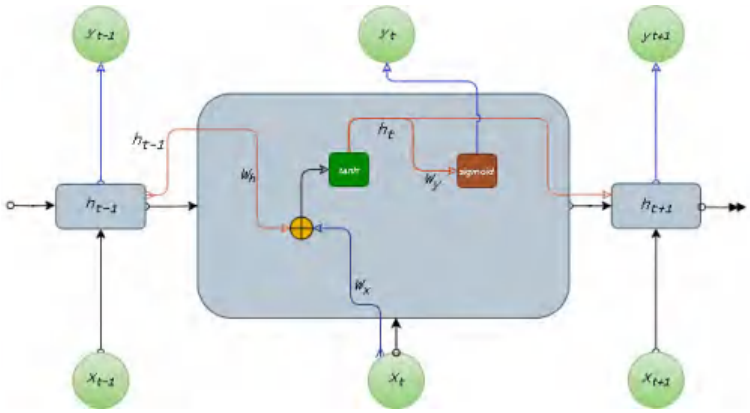
$$\boldsymbol{y}_t = \sigma\left(\boldsymbol{W}_{hy}\boldsymbol{H}_t + \boldsymbol{b}_y\right) \tag{24}$$

dimana:

$\boldsymbol{W}_{xh}, \boldsymbol{W}_{hh}$, dan $\boldsymbol{W}_{hy}$ adalah matriks bobot, $\boldsymbol{H_t}$ merupakan *hidden state* ke-t,
$\boldsymbol{b}_h$ dan $\boldsymbol{b}_y$ adalah bias,
fungsi aktivasi yang umum digunakan adalah $sigmoid$.

Secara teoritis RNN dapat menangkap hubungan jangka panjang, dalam praktiknya RNN sering mengalami masalah *vanishing gradient*, yaitu ketika gradien yang dihitung dalam proses backpropagation melalui waktu (BPTT) menjadi sangat kecil. Akibatnya, bobot pada waktu awal tidak diperbarui secara signifikan, membuat jaringan kesulitan mengingat informasi jangka panjang (Bengio et al., 1994). Hal ini menjadi hambatan utama bagi aplikasi RNN pada data finansial yang memiliki korelasi jangka panjang antar waktu.

Masalah ini secara intuitif disebabkan oleh operasi berulang kali dari transformasi linier dan fungsi aktivasi nonlinier yang menyebabkan gradien eksponensial mengecil. Ketika panjang sekuens meningkat, maka turunan rantai $\frac{\partial L}{\partial W}$ mengandung faktor-faktor $tanh'(z) \approx 1 - tanh^2(z)$ yang berada di bawah satu dan mengecilkan nilai gradien secara progresif.



Gambar 1.8 Arsitektur Dasar (Vanilla) Recurrent Neural Network (RNN)

anishing gradient inilah yang mendorong pengembangan berbagai *ong Short-Term Memory* (LSTM), *Gated Recurrent Unit* (GRU), dan *nit* (SRU) yang lebih stabil dalam menangani dependensi jangka erbaiki efisiensi pelatihan model.

sual mengenai aliran data dan proses internal dalam jaringan RNN a Gambar 1.8 yang menyajikan arsitektur Vanilla RNN yang aimana input pada setiap waktu t tidak hanya diproses secara

langsung, tetapi juga dipengaruhi oleh hidden state dari waktu sebelumnya (H_{t-1}), yang mencerminkan memori jangka pendek dari jaringan. Representasi ini menegaskan struktur rekursif dari RNN dan menjadi dasar bagi pengembangan varian RNN yang lebih kompleks seperti LSTM dan GRU.

b. *Long Short-Term Memory* (LSTM)

1) Struktur LSTM

LSTM merupakan arsitektur RNN yang dirancang untuk mengatasi kelemahan utama dari RNN klasik, yaitu masalah *vanishing gradient* saat mempelajari dependensi jangka panjang. LSTM diperkenalkan oleh (Hochreiter & Schmidhuber, 1997), dan sejak itu menjadi fondasi utama dalam pemodelan data sekuensial yang kompleks seperti sinyal suara, teks, dan time series keuangan.

Fungsi utama dari LSTM didefinisikan oleh persamaan berikut:

$$\boldsymbol{F}_t = \sigma\big(\boldsymbol{W}_f \cdot [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_f\big) \quad \text{(Forget Gate)} \tag{25}$$

$$\boldsymbol{I}_t = \sigma(\boldsymbol{W}_i \cdot [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_i) \quad \text{(Input Gate)} \tag{26}$$

$$\widetilde{\boldsymbol{C}_t} = tanh(\boldsymbol{W}_C \cdot [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_C) \text{ (Candidate Cell State)} \tag{27}$$

$$\boldsymbol{C}_t = \boldsymbol{F}_t \odot \boldsymbol{C}_{t-1} + \boldsymbol{I}_t \odot \widetilde{\boldsymbol{C}_t} \text{ (Cell State Update)} \tag{28}$$

$$\boldsymbol{O}_t = \sigma(\boldsymbol{W}_o \cdot [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_o) \text{ (Output Gate)} \tag{29}$$

$$\boldsymbol{H}_t = \boldsymbol{O}_t \odot tanh(\boldsymbol{C}_t) \text{ (Hidden State)} \tag{30}$$

dengan

$\boldsymbol{X}_t$: input saat waktu t,

$\boldsymbol{H}_t$, $\boldsymbol{H}_{t-1}$: hidden state pada saat t dan waktu sebelumnya $(t-1)$,

σ: fungsi sigmoid,

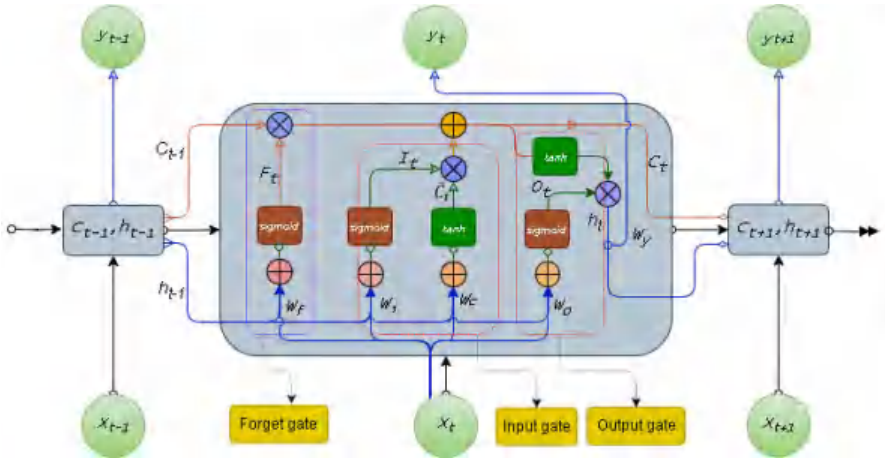
$\odot$: perkalian elemen-per-elemen *(Hadamard Product)*,

$\boldsymbol{C}_t$: state memori internal yang diperbarui setiap waktu.

$\boldsymbol{W}_f, \boldsymbol{W}_i, \boldsymbol{W}_{\tilde{c}}, \boldsymbol{W}_o$: Matriks bobot berturut-turut untuk tahap f*orget, input, candidate cell*, dan *output gate*

$\boldsymbol{b}_f, \boldsymbol{b}_i, \boldsymbol{b}_{\tilde{c}}, \boldsymbol{b}_o$: Vektor bias berturut-turut pada tahap f*orget, input, candidate cell*, dan *output gate*

Pemahaman visual mengenai mekanisme internal Long Short-Term Memory (LSTM) dapat diberika pada Gambar 1.9 yang menyajikan arsitektur standar LSTM. Diagram ini menunjukkan interaksi antara cell state, hidden state, dan tiga jenis gate (input, forget, dan output) dalam satu unit LSTM, yang bekerja secara berurutan pada untuk mengontrol aliran informasi.



Gambar 1.9 Arsitektur Internal Long Short-Term Memory (LSTM)

2) Ukuran Dimensi dan Kompleksitas LSTM

Sebuah LSTM layer terdiri atas empat gerbang utama yaitu ***forget gate***, ***input gate***, ***output gate***, dan ***cell candidate gate***, yang masing-masing memiliki parameter berupa matriks bobot dan bias untuk input saat ini $\boldsymbol{X}_t$ dan *hidden state* sebelumnya $\boldsymbol{H}_{t-1}$ (Goodfellow et al., 2016). Jika m adalah jumlah fitur input (*input size*), p adalah jumlah unit pada *hidden layer* (*hidden size*), n adalah *batch size*, dan T adalah panjang urutan waktu (*time steps*), maka total parameter dalam satu layer LSTM mencakup: bobot input ke hidden $W_f, W_i, W_o, W_c \in R^{p\times m}$ sebanyak $4(pm)$; bobot hidden ke hidden $U_f, U_i, U_o, U_c \in R^{p\times p}$sebanyak $4(p^2)$; serta bias $b_f, b_i, b_o, b_c \in R^p$ sebanyak $4p$. Dengan demikian, total parameter adalah $4(mp + p^2 + p)$. Setiap gate menjalankan dua operasi matriks per langkah waktu: perkalian input $X_t W^T \in R^{n\times p}$ dengan kompleksitas $\mathcal{O}(nmp)$ dan perkalian hidden $H_{t-1}U^T \in R^{n\times p}$ dengan kompleksitas $\mathcal{O}(np^2)$. Karena terdapat empat gate, maka kompleksitas *forward pass* per time step menjadi $\mathcal{O}\big(4np(m+p)\big)$, dan untuk seluruh urutan berdurasi T, kompleksitas total menjadi $\mathcal{O}\big(Tnp(m+p)\big)$, yang menunjukkan bahwa biaya komputasi LSTM tumbuh secara linier terhadap panjang urutan dan batch size, serta kuadrat terhadap ukuran unit tersembunyi.

c. Gated Recurrent Unit (GRU)

1) Struktur Gated Recurrent Unit (GRU)

GRU merupakan varian dari *Recurrent Neural Network* (RNN) yang diperkenalkan oleh Cho et al. (2014) sebagai arsitektur yang lebih efisien dibandingkan LSTM dalam menangani data sekuensial. Tidak seperti LSTM yang memisahkan *cell state* dan *hidden state*, GRU hanya menggunakan satu vektor state $\boldsymbol{H}_t$ dan
nekanisme utama yaitu *update gate* dan *reset gate*. Update gate
apa banyak informasi dari hidden state sebelumnya ($\boldsymbol{H}_{t-1}$) yang
sementara *reset gate* ($\boldsymbol{R}_t$) mengontrol seberapa besar informasi
akan saat membentuk kandidat state baru. Arsitekturnya yang lebih
kit parameter, GRU lebih cepat dilatih dan efisien dalam sumber
i oleh Wan (2023) menunjukkan bahwa GRU mampu menghasilkan

prediksi harga saham jangka pendek yang akurat dengan struktur yang lebih komputasional ringan.

Persamaan kerja internal GRU dinyatakan sebagai berikut:

$$\boldsymbol{Z}_t = \sigma(\boldsymbol{W}_z \cdot [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b}_z) \text{ Update Gate} \quad (31)$$

$$\boldsymbol{R}_t = \sigma(\boldsymbol{W}_r \cdot [\boldsymbol{X}_t, \boldsymbol{H}_{t-1}] + \boldsymbol{b_r}) \text{ Reset Gate} \quad (32)$$

$$\widetilde{\boldsymbol{H}_t} = \tanh(\boldsymbol{W}_h \cdot [\boldsymbol{X}_t, \boldsymbol{R}_t \odot \boldsymbol{H}_{t-1}] + \boldsymbol{b}_h) \text{ Candidate Hidden State} \quad (33)$$

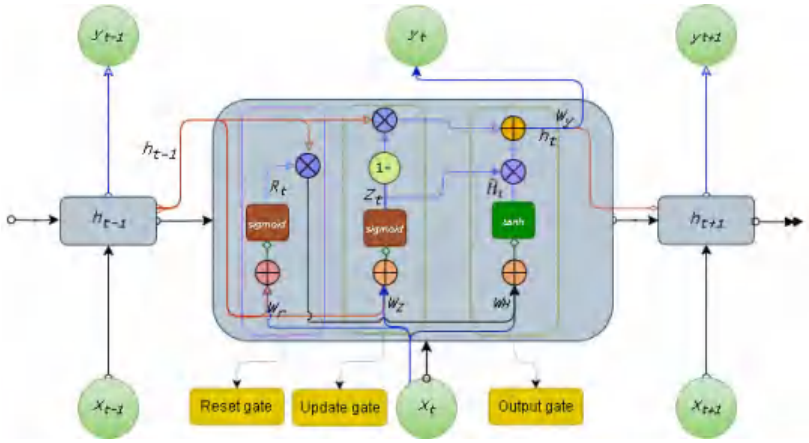
$$\boldsymbol{H}_t = (1 - \boldsymbol{Z}_t) \odot \boldsymbol{H}_{t-1} + \boldsymbol{Z}_t \odot \widetilde{\boldsymbol{H}_t} \text{ Final Hidden State Update} \quad (34)$$

dengan:

$\boldsymbol{X}_t$: input pada waktu t
$\boldsymbol{H}_{t-1}$: *hidden state* pada waktu ke-(*t-1*)
$\boldsymbol{Z}_t, \boldsymbol{R}_t$: *update* dan reset gate pada waktu t
$\widetilde{\boldsymbol{H}_t}$: kandidat *hidden state* pada waktu t
$\boldsymbol{W}_z, \boldsymbol{W}_r, \boldsymbol{W}_h$: bobot koneksi
$\boldsymbol{b}_z, \boldsymbol{b}_r, \boldsymbol{b}_h$: bias masing-masing
σ : fungsi aktivasi sigmoid
$tanh$: fungsi aktivasi hiperbolik
$\odot$: perkalian elemen-per-elemen

GRU merupakan varian ringan dari LSTM dengan arsitektur dua gerbang—update dan reset—yang menyatukan *hidden state* dan *cell state* menjadi satu vektor memori. Struktur ini mengurangi kompleksitas dan jumlah parameter, menghasilkan pelatihan yang lebih cepat dan efisien tanpa kehilangan kemampuan dalam menangkap dependensi jangka panjang (Chung et al., 2014). Dalam konteks prediksi harga saham, GRU menunjukkan performa kompetitif dengan akurasi yang stabil serta efisiensi tinggi, terutama pada dataset yang fluktuatif (Shejul et al., 2023). Kombinasi GRU dengan metode dekomposisi atau model statistik seperti ARIMA juga menjanjikan peningkatan akurasi prediksi jangka pendek (C. Wei et al., 2025).

Gambar 1.10 menyajikan visualisasi arsitektur GRU standar sebagai perjelasan alur kerja internal GRU. Diagram ini menampilkan peran masing-masing gerbang dalam mengatur informasi dari input dan hidden state sebelumnya, serta proses pembaruan hidden state. Visualisasi ini memberikan pemahaman intuitif sebelum membandingkannya dengan arsitektur LSTM dan SRU pada bagian berikutnya.



Gambar 1.10 Arsitektur Internal Gated Recurrent Unit (GRU)

2) Ukuran Dimensi dan Kompleksitas GRU

GRU memiliki struktur parameter yang lebih ringkas dibandingkan LSTM, dengan total parameter sebanyak $3(mp + p^2 + p)$, mencakup bobot input, bobot rekuren, dan bias dari tiga komponen utama: update gate, reset gate, dan kandidat hidden state. Dimensi input $\boldsymbol{X_t} \in R^{n \times m}$, dan hidden state $\boldsymbol{H}_t \in R^{n \times p}$, di mana n adalah ukuran batch, m adalah jumlah fitur input, dan p adalah jumlah unit tersembunyi. Kompleksitas komputasi per time step adalah $\mathcal{O}(nmp + np^2)$, dan secara keseluruhan per sequence berdurasi T langkah menjadi $\mathcal{O}\big(Tn(mp + p^2)\big)$, sesuai dengan formulasi arsitektural yang diuraikan oleh (Goodfellow et al., 2016). Efisiensi ini menjadikan GRU cocok untuk pemrosesan sekuensial dengan biaya komputasi yang lebih rendah dibanding LSTM, sambil tetap mempertahankan kemampuan dalam menangkap dependensi temporal jangka panjang (Chung et al., 2014).

d. Simple Recurrent Unit (SRU)

1) Struktur Simple Recurrent Unit (SRU)

SRU adalah salah satu varian dari RNN yang dirancang untuk mempercepat komputasi dengan mengurangi dependensi berurutan (*sequential dependency*) tanpa mengorbankan performa prediksi. SRU dikembangkan oleh (Lei et al., 2018) sebagai solusi atas keterbatasan LSTM dan GRU dalam hal kecepatan komputasi dan efisiensi paralelisasi.

SRU memisahkan antara transformasi input dan propagasi status (*state propagation*), memungkinkan sebagian besar operasi dilakukan secara paralel. Tidak seperti LSTM dan GRU yang menghitung gate dengan melibatkan hidden state sebelumnya ($\boldsymbol{H}_{t-1}$), SRU hanya menggunakan *cell state* sebelumnya ($\boldsymbol{C}_{t-1}$) dalam adikannya lebih ringan secara komputasi.

tis SRU

dua gerbang utama: *forget gate* ($\boldsymbol{F}_t$) dan *reset gate* ($\boldsymbol{R}_t$), serta dan *hidden state* ($\boldsymbol{H}_t$). Berikut adalah formulasi matematisnya:

$$\boldsymbol{G}_t = \boldsymbol{X}_t \boldsymbol{W} + \boldsymbol{b} \tag{35}$$

$$\boldsymbol{F}_t = \sigma\left(\boldsymbol{X}_t\boldsymbol{W}_f + \boldsymbol{C}_{t-1} \odot \boldsymbol{V}_f + \boldsymbol{b}_f\right) \quad (36)$$

$$\boldsymbol{R}_t = \sigma(\boldsymbol{X}_t\boldsymbol{W}_r + \boldsymbol{C}_{t-1} \odot \boldsymbol{V}_r + \boldsymbol{b}_r) \quad (37)$$

$$\boldsymbol{C}_t = \boldsymbol{F}_t \odot \boldsymbol{C}_{t-1} + (1 - \boldsymbol{F}_t) \odot \boldsymbol{G}_t \quad (38)$$

$$\boldsymbol{H}_t = \boldsymbol{R}_t \odot \boldsymbol{C}_t + (1 - \boldsymbol{R}_t) \odot \boldsymbol{X}_t\, \boldsymbol{W}_{rescale} \quad (39)$$

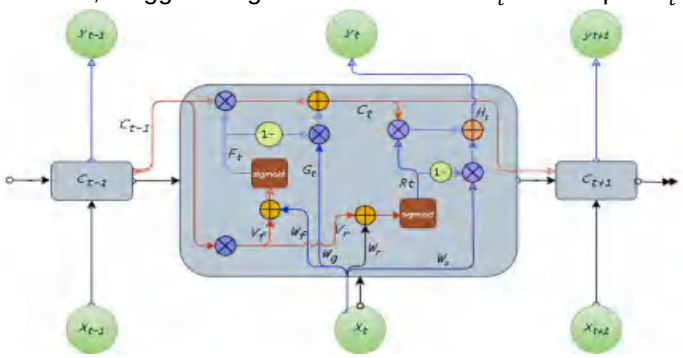
dimana

$\boldsymbol{X}_t$: input multivariabel pada *time step* t, dengan:
$\boldsymbol{C}_t$: *cell state* pada *time step* t,
$\boldsymbol{H}_t$: *hidden state* (output) pada time step t,
$\boldsymbol{W}, \boldsymbol{W}_f, \boldsymbol{W}_r, \boldsymbol{W}_{rescale}$: bobot input ke hidden,
$\boldsymbol{V}_f, \boldsymbol{V}_r$: vektor bobot untuk koneksi state (*broadcastable*),
$\boldsymbol{b}, \boldsymbol{b}_f, \boldsymbol{b}_r$: bias.

SRU menawarkan keunggulan utama dalam hal efisiensi komputasi karena proses gating-nya hanya bergantung pada cell state sebelumnya, bukan hidden state, sehingga mendukung paralelisasi penuh di seluruh time step. Arsitektur ini terbukti lebih cepat dari LSTM dan GRU tanpa kehilangan performa prediksi, terutama pada data sekuensial panjang (Lei et al., 2018). Dengan jumlah parameter yang lebih sedikit, SRU memberikan hasil yang kompetitif dalam pemodelan deret waktu.

Ruang pengembangan SRU mencakup integrasi dengan attention mechanism, penggabungan dalam model hybrid seperti SRU-LSTM atau SRU-CNN, serta tuning hyperparameter menggunakan algoritma seperti Bayesian Optimization dan PSO. Penambahan layer normalization dan dropout juga menjadi strategi penting untuk meningkatkan stabilitas dan generalisasi model (Ba et al., 2016; Srivastava et al., 2014).

Diagram arsitektur SRU pada satu time step t disajikan pada Gambar 1.11 yang menampilkan bagaimana input multivariat $\boldsymbol{X}_t$ diproses melalui transformasi linier, gate forget dan reset, hingga menghasilkan cell state $\boldsymbol{C}_t$ dan output $\boldsymbol{H}_t$.



1.11 Arsitektur Internal Simple Recurrent Unit (SRU)

dan Kompleksitas SRU

nsi dan kompleksitas komputasi SRU dirancang untuk efisiensi san data sekuensial. Dengan input $\boldsymbol{X} \in R^{n \times T \times m}$ dan output hidden RU hanya bergantung pada cell state sebelumnya $\boldsymbol{C}_{t-1}$ tanpa state $\boldsymbol{H}_{t-1}$, sehingga memungkinkan paralelisasi penuh antar time

step. Kompleksitas komputasi per langkah waktu sebesar $\mathcal{O}(n \cdot m \cdot p)$, yang lebih efisien dibandingkan LSTM dan GRU. Secara total, SRU memiliki parameter sebanyak $4 \times (m \cdot p + p)$, mencakup transformasi kandidat cell $\boldsymbol{W}$, gate forget $\boldsymbol{W}_f, \boldsymbol{V}_f, \boldsymbol{b}_f$, gate reset $\boldsymbol{W}_r, \boldsymbol{V}_r, \boldsymbol{b}_r$, serta proyeksi shortcut $\boldsymbol{W}_{rescale}$ dan bias output. Struktur ini menjadikan SRU kompetitif dalam performa sekaligus unggul dalam efisiensi memori dan waktu pelatihan pada perangkat GPU (Lei et al., 2018).

1.7.5 Model Ensembel dan Stacked RNN

Stacking atau arsitektur berlapis (*stacked architecture*) dalam *deep learning* merujuk pada proses menyusun beberapa lapisan jaringan secara vertikal sehingga memungkinkan pembelajaran representasi kompleks dari data. Dalam konteks RNN, stacking dilakukan dengan **menyusun beberapa lapisan RNN secara berurutan**, di mana output dari satu lapisan digunakan sebagai input untuk lapisan berikutnya, meningkatkan kapasitas model dalam menangkap dinamika temporal nonlinier dan hierarkis (Pascanu et al., 2013; Tuan et al., 2022).

Secara umum, jika diberikan input sekuensial $\boldsymbol{X} = \{x_1, x_2, \dots, x_T\}$, maka pada lapisan pertama ($l = 1$):

$$h_t^{(1)} = \mathcal{F}^{(1)}\left(x_t, h_{t-1}^{(1)}; \theta^{(1)}\right) \tag{40}$$

Untuk lapisan ke-l ($l = 2, \dots, L$), representasi diperbarui sebagai:

$$h_t^{(l)} = \mathcal{F}^{(\ell)}\left(h_t^{(l-1)}, h_{t-1}^{(l)}; \theta^{(l)}\right) \tag{41}$$

di mana:

$\mathcal{F}^{(\ell)}$ adalah fungsi RNN (LSTM, GRU, SRU) pada lapisan ke-l,
$\theta^{(l)}$ adalah parameter (berbobot) pada lapisan ke-l,
$h_t^{(l)} \in R^p$ adalah *hidden state* pada waktu t dan lapisan ke-l.

Output akhir jaringan *stacked*-RNN biasanya diambil dari hidden state lapisan paling atas:

$$\hat{y_t} = \sigma_o\left(h_t^{(L)} \cdot W_o + b_o\right) \tag{42}$$

dengan σ sebagai fungsi aktivasi (mis. identitas untuk regresi, sigmoid/softmax untuk klasifikasi).

Keunggulan *stacking* terletak pada kemampuannya membentuk representasi multilevel. Lapisan bawah dapat menangkap dinamika jangka pendek, sedangkan lapisan atas belajar dependensi jangka panjang dan pola konseptual tingkat tinggi (Lecun et al., 2015). *Stacking* juga memperbesar kapasitas ekspresif model, memungkinkan
n nonlinear kompleks yang sulit ditangkap oleh model dangkal.
kedalaman jaringan membawa tantangan seperti *vanishing*
ısi kinerja saat pelatihan, yang diatasi dengan penggunaan teknik
ayer normalization, *dropout*, atau *pretraining layer-wise* (Kaiming et
onteks time series, *stacking* telah diadopsi untuk meningkatkan
m prediksi saham, cuaca, dan beban listrik, di mana data bersifat
uh fluktuasi.

1.7.6 Optimasi Hyperparameter dalam Model Deep Learning

Tuning hyperparameter merupakan tahap krusial dalam pengembangan model deep learning, terutama untuk arsitektur kompleks seperti LSTM, GRU, dan SRU yang digunakan dalam pemodelan data deret waktu. Tidak seperti parameter internal model yang dipelajari selama proses pelatihan, hyperparameter ditentukan sebelum pelatihan dan secara langsung memengaruhi dinamika pembelajaran serta kemampuan generalisasi model (Bergstra & Bengio, 2012). Beberapa hyperparameter penting yang sering digunakan meliputi ukuran unit tersembunyi (*hidden units*), tingkat pembelajaran (*learning rate*), ukuran batch, jumlah *epoch*, *dropout rate*, serta konfigurasi arsitektural seperti jumlah layer dan aktivasi.

Dalam konteks forecasting multivariabel dengan data finansial, tuning parameter menjadi semakin penting karena dinamika pasar yang bersifat nonstasioner dan kompleks menuntut model untuk beradaptasi terhadap pola yang berubah-ubah. Oleh karena itu, strategi tuning yang sistematis seperti *random search*, *Bayesian optimization*, atau metaheuristik (misalnya PSO dan GWO) menjadi penting untuk menemukan konfigurasi optimal. Pendekatan ini tidak hanya bertujuan memaksimalkan akurasi prediksi (misalnya melalui penurunan RMSE atau peningkatan R^2), tetapi juga meningkatkan efisiensi pelatihan dan stabilitas model selama validasi silang (Feurer & Hutter, 2019; Franceschi et al., 2017).

GWO, yang diusulkan oleh (Mirjalili et al., 2014a), mengadopsi perilaku sosial dan strategi berburu serigala abu-abu dalam mengelilingi, mengejar, dan menyerang mangsa untuk mengoptimasi fungsi objektif. Dalam konteks tuning hyperparameter, GWO menunjukkan keunggulan dalam menghindari jebakan lokal dan konvergensi cepat, menjadikannya alternatif kompetitif bagi PSO, terutama pada permasalahan yang memerlukan eksplorasi ruang pencarian yang luas dan nonkonveks (Mohakud & Dash, 2022).

Praktiknya, metode otomatis seperti GWO dapat diintegrasikan dalam pipeline pelatihan model untuk menyesuaikan parameter seperti jumlah unit tersembunyi, learning rate, batch size, dropout rate, dan jumlah layer. Dengan meningkatnya kompleksitas model dan ukuran data, tuning otomatis menjadi komponen penting dalam pengembangan sistem prediktif yang tidak hanya akurat, tetapi juga efisien dan adaptif terhadap dinamika data.

a. Random Search: keunggulan dan keterbatasan

Random Search merupakan metode pencarian hyperparameter yang sederhana namun efektif, dengan pendekatan sampling acak dari ruang pencarian. Diperkenalkan secara formal oleh (Bergstra & Bengio, 2012), metode ini menghindari kelemahan eksplorasi statis pada grid search dengan melakukan sampling dari distribusi [illegible] ntukan atas setiap dimensi hyperparameter.

s

erparameter $\mathcal{H} = \mathcal{H}_1 \times \mathcal{H}_2 \times \cdots \times \mathcal{H}_d$, dengan d adalah jumlah $\mathcal{H}_i$ sebagai domain untuk hyperparameter ke-i. Tujuan *random* mukan konfigurasi $\theta^* \in \mathcal{H}$ yang meminimalkan fungsi loss $\mathcal{L}(\theta)$,

$$\theta^* = \arg\min_{\theta \in \mathcal{H}} \mathcal{L}(\theta)$$

Random Search memilih N kandidat $\theta^{(1)}, \ldots, \theta^{(N)} \sim \text{Uniform}(\mathcal{H})$, lalu mengevaluasi $\mathcal{L}(\theta^{(i)})$ untuk masing-masing. Solusi terbaik adalah:

$$\theta^* = \theta^{(i^*)} \text{ dengan } i^* = \arg\min_{1 \le i \le N} \mathcal{L}(\theta^{(i)})$$

Keunggulan utama random search terletak pada sifat eksploratif dan efisiensinya. Tidak seperti *grid search* yang mengevaluasi semua titik secara terstruktur, random search memiliki peluang lebih besar untuk menemukan konfigurasi optimal dengan jumlah evaluasi yang lebih sedikit, terutama ketika hanya sebagian kecil dari parameter yang berdampak signifikan terhadap performa. Selain itu, metode ini mudah diimplementasikan, tidak memerlukan asumsi khusus terhadap fungsi objektif, serta dapat dijalankan secara paralel tanpa saling ketergantungan antar iterasi (Feurer & Hutter, 2019)

Algoritma 1. Pseudocode Random Search

```
Input:
  - Space of hyperparameters H = {h₁, h₂, ..., h_d}
  - Number of iterations N
  - Objective function L(θ): validation loss for configuration θ

Output:
  - Best hyperparameter configuration θ*

Algorithm:
1. Initialize θ_best ← null
2. Initialize L_best ← ∞
3. For i = 1 to N do:
     a. Sample θ_i ~ Uniform(H)
     b. Evaluate loss_i ← L(θ_i)
     c. If loss_i < L_best then:
           i. L_best ← loss_i
           ii. θ_best ← θ_i
4. Return θ_best
```

ikian, random search memiliki **keterbatasan adaptif**. Karena ecara acak dan tanpa mempertimbangkan informasi dari evaluasi roses pencarian tidak diarahkan secara cerdas ke wilayah ruang anjikan. Hal ini menjadikannya kurang optimal dibanding metode *sian Optimization* atau algoritma evolusioner ketika diperlukan ang tinggi (Claesen & De Moor, 2015). Selain itu, hasil pencarian

sangat dipengaruhi oleh bagaimana peneliti menentukan rentang dan distribusi sampling hyperparameter, sehingga memerlukan intuisi awal yang baik.

Meskipun demikian, random search tetap menjadi baseline yang kuat dan dapat diandalkan, khususnya untuk eksperimen awal eksplorasi konfigurasi model RNN dan LSTM dalam pemodelan deret waktu finansial, di mana fleksibilitas dan kemudahan penerapannya menjadi nilai tambah utama

b. Grey Wolf Optimizer (GWO): inspirasi perilaku sosial

Grey Wolf Optimizer (GWO) adalah algoritma optimasi metaheuristik yang diperkenalkan oleh (Mirjalili et al., 2014a), yang meniru perilaku sosial dan strategi berburu serigala abu-abu di alam liar. GWO mensimulasikan mekanisme sosial hierarkis kawanan serigala — terdiri dari empat peran utama: alfa (pemimpin), beta (wakil), delta (pengikut senior), dan omega (pengikut biasa). Dalam konteks optimasi, solusi terbaik sejauh ini direpresentasikan oleh serigala alfa, sedangkan dua solusi terbaik berikutnya diwakili oleh beta dan delta.

Dalam optimasi hyperparameter untuk deep learning, GWO terbukti efektif karena pendekatannya yang seimbang antara eksplorasi global dan eksploitasi lokal. Strategi ini memungkinkan pencarian konfigurasi optimal secara efisien, bahkan dalam ruang parameter yang kompleks dan non-konveks seperti jumlah neuron tersembunyi, learning rate, dan dropout rate pada arsitektur seperti LSTM dan GRU (Gupta, Gupta, et al., 2023).

Rumusan Matematis

GWO mengupdate posisi tiap serigala (solusi) berdasarkan posisi tiga pemimpin terbaik $\left(\overrightarrow{X_\alpha}, \overrightarrow{X_\beta}, \overrightarrow{X_\delta}\right)$:

$$\overrightarrow{D_\alpha} = \left|\overrightarrow{C_1} \cdot \overrightarrow{X_\alpha} - \vec{X}\right|, \quad \overrightarrow{D_\beta} = \left|\overrightarrow{C_2} \cdot \overrightarrow{X_\beta} - \vec{X}\right|, \quad \overrightarrow{D_\delta} = \left|\overrightarrow{C_3} \cdot \overrightarrow{X_\delta} - \vec{X}\right|$$

$$\overrightarrow{X_1} = \overrightarrow{X_\alpha} - \overrightarrow{A_1} \cdot \overrightarrow{D_\alpha}, \quad \overrightarrow{X_2} = \overrightarrow{X_\beta} - \overrightarrow{A_2} \cdot \overrightarrow{D_\beta}, \quad \overrightarrow{X_3} = \overrightarrow{X_\delta} - \overrightarrow{A_3} \cdot \overrightarrow{D_\delta}$$

$$\vec{X}(t+1) = \frac{1}{3}\left(\overrightarrow{X_1} + \overrightarrow{X_2} + \overrightarrow{X_3}\right)$$

dengan $\vec{A} = 2\vec{a} \cdot \vec{r} - \vec{a}$ dan $\vec{C} = 2 \cdot \vec{r}$, di mana $\vec{a}$ menurun secara linear dari 2 ke 0 selama iterasi, dan $\vec{r} \sim \mathcal{U}(0,1)$.

Keunggulan utama GWO meliputi struktur yang sederhana, kemudahan implementasi, efisiensi konvergensi, serta kesesuaian untuk fungsi objektif yang tidak itu, sifatnya yang bebas gradien menjadikan GWO sangat fleksibel deep learning dengan arsitektur kompleks seperti LSTM–GRU 2023). Namun, GWO juga memiliki **keterbatasan** seperti potensi ır ketika keragaman populasi rendah, serta kesulitan dalam ıensi sangat tinggi tanpa strategi peningkatan eksplorasi adaptif.

Algoritma 2. Pseudocode GWO

```
Input:
  - Fungsi objektif f(X)
  - Jumlah serigala N, iterasi maksimal T

Output:
  - Posisi terbaik X_α

1. Inisialisasi posisi semua serigala X_i secara acak
2. Evaluasi f(X_i) untuk seluruh serigala
3. Tentukan tiga posisi terbaik: X_α, X_β, X_δ
4. Untuk t = 1 hingga T:
     a. Untuk setiap serigala i:
         i. Hitung A, C untuk α, β, δ
         ii. Update posisi X_i berdasarkan rata-rata X_1, X_2, X_3
     b. Evaluasi f(X_i)
     c. Update X_α, X_β, X_δ jika diperlukan
5. Kembalikan X_α
```

c. Definisi Paramater

1) Learning rate

Learning rate adalah hyperparameter utama yang mengontrol seberapa besar pembaruan bobot dilakukan dalam setiap langkah optimisasi selama proses pelatihan model neural network. Nilai *learning rate* yang terlalu besar dapat menyebabkan pelatihan tidak konvergen karena melompati minimum global, sedangkan nilai yang terlalu kecil menyebabkan pelatihan menjadi lambat dan berpotensi terjebak pada minimum lokal. Secara matematis, pembaruan bobot dilakukan dengan rumus $w_{t+1} = w_t - \eta \cdot \nabla \mathcal{L}(w_t)$ di mana η adalah *learning rate* dan $\mathcal{L}$ adalah gradien dari fungsi loss (Goodfellow et al., 2016).

2) Batch size

Batch size adalah jumlah sampel yang diproses sekaligus dalam satu iterasi pelatihan sebelum parameter model diperbarui. *Batch size* mempengaruhi stabilitas dan
n. Batch kecil cenderung mempercepat update parameter namun
(*noisy*), sedangkan batch besar menghasilkan estimasi gradien
un membutuhkan sumber daya komputasi yang lebih besar (Keskar
an *batch size* juga berinteraksi dengan *learning rate*, di mana
penting untuk mencapai generalisasi yang baik.

3) Dropout rate

Dropout rate adalah probabilitas untuk secara acak menonaktifkan unit neuron selama pelatihan, guna mencegah *overfitting* dan meningkatkan generalisasi model. Dropout mencegah ko-adaptasi antar neuron dan bertindak sebagai regularisasi struktural. Jika *dropout rate* adalah p, maka selama pelatihan setiap neuron diaktifkan dengan probabilitas $1 - p$. Teknik ini pertama kali diperkenalkan oleh (Srivastava et al., 2014) dan telah menjadi praktik umum dalam pelatihan deep learning.

4) Epoch

Epoch adalah satu siklus penuh di mana seluruh dataset pelatihan digunakan sekali untuk memperbarui bobot model. Jumlah *epoch* memengaruhi seberapa lama model belajar terhadap data. Terlalu sedikit *epoch* menyebabkan underfitting karena model belum belajar optimal, sedangkan terlalu banyak *epoch* dapat menyebabkan *overfitting* terhadap data pelatihan. Biasanya, pemilihan *epoch* dikombinasikan dengan strategi *early stopping* untuk mencegah pelatihan berlebihan (Prechelt, 1998).

5) Jumlah Layer dan Unit Neuron

Proses perancangan model jaringan saraf dalam penelitian ini, dua komponen utama yang diperhatikan secara eksplisit adalah jumlah layer (*depth*) dan jumlah unit neuron (*width*) pada setiap layer tersembunyi. Meskipun keduanya sering disebut bersama-sama dalam konfigurasi arsitektural, secara konseptual keduanya merupakan entitas yang berbeda dan memiliki implikasi yang spesifik terhadap kapasitas representasi model.

Jumlah layer merujuk pada banyaknya lapisan yang menyusun jaringan saraf, yang mencakup *hidden layer* dan *output layer*, sementara input layer tidak dihitung karena tidak memiliki parameter pelatihan. Semakin dalam jaringan (dalam hal jumlah layer tersembunyi), semakin tinggi kemampuan model dalam menangkap representasi hierarkis dari fitur data secara bertahap. Arsitektur dengan lebih dari satu layer tersembunyi disebut sebagai deep neural network. Walau demikian, peningkatan jumlah layer juga membawa konsekuensi seperti meningkatnya risiko *vanishing gradient*, *overfitting*, dan kompleksitas komputasi, khususnya jika tidak didukung oleh teknik normalisasi, residual connection, atau regulasi lainnya (Goodfellow et al., 2016).

Sementara itu, jumlah unit neuron mengacu pada banyaknya neuron yang terdapat dalam satu layer tertentu. Setiap neuron berfungsi sebagai unit komputasi yang mentransformasikan input dari layer sebelumnya melalui kombinasi linier (menggunakan bobot dan bias) dan fungsi aktivasi nonlinier. Jumlah neuron menentukan kapasitas informasi yang dapat diproses pada setiap langkah transformasi. Model dengan neuron
ada setiap layer umumnya memiliki kapasitas representasi yang
ga lebih rentan terhadap overfitting, terutama dalam kasus jumlah
erbatas (Srivastava et al., 2014). Oleh karena itu, dalam eksperimen
ılah neuron diuji secara sistematis melalui proses tuning
k mendapatkan keseimbangan antara *underfitting* dan *overfitting*.
hal tersebut, pemilihan jumlah layer dan jumlah neuron per layer
aruhi kompleksitas dan kedalaman jaringan, tetapi juga berdampak

langsung terhadap performa prediktif dan kestabilan generalisasi model. Kombinasi optimal antara keduanya ditentukan melalui proses tuning berbasis pendekatan metaheuristik dan validasi kinerja model prediksi harga saham yang digunakan dalam penelitian ini.

1.7.7 Normalisasi Data

Normalisasi data merupakan tahap penting dalam praproses yang bertujuan untuk menyamakan skala numerik antar fitur input sebelum dimasukkan ke dalam model deep learning. Dalam konteks prediksi harga saham menggunakan RNN, data historis seperti harga (*open*, *close*, *high*, *low*) dan volume perdagangan biasanya memiliki skala yang berbeda secara signifikan. Perbedaan ini dapat menyebabkan dominasi fitur tertentu dalam proses pembelajaran dan memperlambat konvergensi selama pelatihan model (Gopal et al., 2015).

Metode normalisasi yang umum digunakan dalam penelitian ini adalah **Min-Max Scaling**, yang mentransformasi setiap nilai fitur ke dalam rentang [0,1] menggunakan rumus:

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \tag{43}$$

Pendekatan ini mempertahankan distribusi asli data namun menghindari nilai ekstrem yang bisa memicu ledakan gradien dalam jaringan berulang. Untuk menjaga validitas evaluasi, proses normalisasi hanya dilakukan berdasarkan statistik dari **data latih**, lalu diterapkan pada data validasi dan data uji secara konsisten (Jain et al., 2000). Praktik ini mencegah terjadinya *data leakage*, yang dapat mengakibatkan performa model tampak lebih baik dari sebenarnya.

Metode selain Min-Max, beberapa eksperimen juga dilakukan dengan pendekatan **Z-Score Normalization** atau **Standardization**, terutama pada fitur volume atau indikator teknikal seperti RSI dan MACD yang memiliki penyebaran nilai lebih besar. Dalam hal ini, data ditransformasikan agar memiliki rata-rata nol dan deviasi standar satu:

$$x' = \frac{x - \mu}{\sigma} \tag{44}$$

Pemilihan metode normalisasi bergantung pada karakteristik fitur dan sensitivitas arsitektur model terhadap skala data. Dalam penelitian ini, kombinasi Min-Max digunakan untuk fitur harga, sedangkan standardisasi diterapkan pada indikator turunan berbasis osilator. Hasil evaluasi empiris menunjukkan bahwa pemisahan strategi normalisasi berdasarkan jenis fitur memberikan performa prediktif yang lebih stabil pada model stacked LSTM–GRU.

1.7.8 Evaluasi Kinerja Model Prediksi

Evaluasi model prediksi merupakan komponen penting dalam menilai kualitas model regresi atau peramalan. Berbagai metrik digunakan untuk rediksi model terhadap data aktual. Salah satu metrik paling umum **quared Error (RMSE)**, yang mengukur seberapa besar rata-rata nilai aktual y_t dan nilai prediksi $\hat{y}_t$. RMSE sangat sensitif terhadap na pemberian penalti kuadrat, dan dirumuskan sebagai berikut:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \widehat{y_t})^2} \quad (45)$$

RMSE berguna ketika model perlu menekankan ketepatan absolut dan meminimalkan kesalahan besar, meskipun kelemahannya terletak pada ketidakdiskalaan sehingga sulit dibandingkan antar dataset berbeda (Hyndman & Athanasopoulos, 2018).

Kegunaan dari **koefisien determinasi** atau **R-squared *(R^2)*** yaitu untuk mengukur seberapa besar variasi dalam data aktual yang dapat dijelaskan oleh model. Semakin besar nilai R^2, semakin baik model dalam menjelaskan fluktuasi data. Nilainya berkisar antara 0 dan 1, dengan interpretasi R^2= 1 menunjukkan prediksi sempurna. *R²* dihitung menggunakan formula berikut:

$$R^2 = 1 - \frac{\sum_{t=1}^{n}(y_t - \widehat{y_t})^2}{\sum_{t=1}^{n}(y_t - \bar{y})^2} \quad (46)$$

di mana $\bar{y}$ adalah rata-rata nilai aktual. Jika nilai R^2 negatif, itu menandakan bahwa model lebih buruk daripada sekadar menggunakan rata-rata nilai sebagai prediksi (Draper & Smith, 1998).

Sementara itu, untuk mengukur akurasi dalam bentuk relatif atau persentase, digunakan **Mean Absolute Percentage Error (MAPE)**. MAPE sangat populer dalam bidang bisnis dan ekonomi karena hasilnya mudah dipahami oleh non-teknisi. MAPE menghitung rata-rata dari kesalahan absolut dalam bentuk persentase terhadap nilai aktual, dengan rumus:

$$\text{MAPE} = \frac{100\%}{n}\sum_{t=1}^{n}\left|\frac{y_t - \widehat{y_t}}{y_t}\right| \quad (47)$$

Kelebihan MAPE adalah hasil yang bersifat diskalakan, tetapi kelemahannya muncul ketika terdapat nilai aktual yang mendekati nol, sehingga menyebabkan distorsi besar pada nilai error (Makridakis et al., 2018).

Untuk mengatasi bias MAPE terhadap outlier dan memberikan penalti lebih besar terhadap error besar, digunakan metrik alternatif yaitu **Root Mean Squared Percentage Error (RMSPE)**. Metrik ini mengkuadratkan persentase error terlebih dahulu sebelum dihitung rata-ratanya dan diakarkan. Rumus matematis RMSPE adalah sebagai berikut:

$$\text{RMSPE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(\frac{y_t - \widehat{y_t}}{y_t}\right)^2} \times 100\% \quad (48)$$

RMSPE sering digunakan dalam domain keuangan dan prediksi pasar yang memiliki data volatil, karena kemampuannya untuk menangkap proporsi error besar dalam bentuk persentase yang lebih presisi (Lawi et al., 2022a).

Secara keseluruhan, pemilihan metrik evaluasi yang tepat sangat bergantung
ısalahan dan karakteristik data. Penggunaan kombinasi metrik
APE, dan RMSPE memberikan gambaran lebih menyeluruh atas
ıtu model.
ıat metrik tersebut, metrik tambahan yang direkomendasikan dalam
lain adalah **Nash-Sutcliffe Efficiency (NSE)** dan **Percent Bias**
evaluasi efisiensi model prediksi dengan membandingkan model
aktual:

$$\mathrm{NSE} = 1 - \frac{\sum_{i=1}^{n}(O_i - \widehat{O_i})^2}{\sum_{i=1}^{n}(O_i - \bar{O})^2} \quad (49)$$

NSE digunakan secara luas dalam studi hidrologi dan ekologi serta mulai diadopsi pada time series keuangan dan multivariat (Samantaray & Sahoo, 2024a). Nilai NSE mendekati 1 menunjukkan performa sangat baik, sedangkan nilai < 0 menandakan prediksi lebih buruk dari sekadar mengambil rata-rata historis.

Percent Bias (PBIAS) menunjukkan kecenderungan model dalam memperkirakan nilai yang lebih tinggi atau lebih rendah dari nilai aktual:

$$\mathrm{PBIAS} = 100\% \times \frac{\sum_{t=1}^{n}(y_t - \widehat{y_t})}{\sum_{t=1}^{n} y_t} \quad (50)$$

PBIAS positif menandakan model *underestimation*, sedangkan negatif menandakan *overestimation* (Samantaray et al., 2025a). Metrik ini berguna sebagai indikator bias sistematis.

Sebagai pelengkap, *Willmott's Index of Agreement* (WI). WI pertama kali diperkenalkan oleh Willmott (1981) sebagai alternatif terhadap koefisien determinasi (R-kuadrat), dengan tujuan memberikan ukuran kesesuaian (*agreement*) antara nilai aktual dan prediksi yang lebih sensitif terhadap perbedaan absolut. WI menilai seberapa dekat hasil prediksi terhadap nilai observasi berdasarkan deviasi absolut terhadap nilai rata-rata aktual. Secara matematis, WI dirumuskan sebagai berikut:

$$\mathrm{WI} = 1 - \frac{\sum_{i=1}^{n}(y_i - \widehat{y_i})^2}{\sum_{i=1}^{n}(|\widehat{y_i} - \bar{y}| + |y_i - \bar{y}|)^2} \quad (51)$$

dengan y_i menyatakan nilai aktual, $\widehat{y_i}$ adalah hasil prediksi, dan $\bar{y}$ merupakan rata-rata dari nilai aktual. Nilai WI berada dalam rentang 0 hingga 1, di mana nilai mendekati 1 menunjukkan tingkat kesesuaian yang tinggi antara hasil prediksi dan observasi aktual, sedangkan nilai mendekati 0 menunjukkan bahwa model tidak lebih baik dari sekadar menggunakan nilai rata-rata sebagai prediksi (Samantaray et al., 2025a). Penggunaan WI dalam penelitian ini didasarkan pada keunggulannya dalam memberikan penilaian menyeluruh terhadap kemampuan model dalam merepresentasikan pola data, khususnya pada data deret waktu seperti harga saham yang memiliki karakteristik volatil dan nonlinier.

Implementasi RNN ini termasuk arsitektur *Long Short-Term Memory* (LSTM), *Gated Recurrent Unit* (GRU), dan *Simple Recurrent Unit* (SRU), WI memberikan ukuran yang lebih menyeluruh terhadap performa model karena mempertimbangkan fluktuasi nilai prediksi relatif terhadap rata-rata nilai aktual. Oleh karena itu, WI menjadi pelengkap penting dalam evaluasi performa selain metrik lain seperti *RMSE, MAPE, R-squred*, dan *NSE*. Penggunaan *WI* secara simultan dengan metrik lainnya memungkinkan evaluasi yang lebih objektif terhadap kesesuaian model dalam menangkap dinamika pasar saham.

BAB II
PENDEKATAN MULTIVARIABEL DEEP LEARNING: ANSAMBEL STACKED-LSTM DENGAN FITUR REKAYASA

2.1 Abstrak

Latar belakang. Studi ini menganalisis karakteristik unik saham dan dinamika berbagai ekuitas yang menjadi daya tarik utama bagi investor karena potensi pengembalian yang tinggi. Volatilitas harga saham membuka peluang strategis bagi investor yang mampu mengantisipasi arah pergerakan pasar secara tepat. Untuk mendukung hal tersebut, dibutuhkan metodologi prediktif yang efektif dalam mengenali pola deret waktu dan relasi antar fitur. **Tujuan.** Penelitian ini mengembangkan pendekatan prediksi harga saham multivariabel menggunakan model *Long Short-Term Memory* (LSTM) dalam konfigurasi bertingkat (*stacked*), dengan skema input tunggal dan input paralel. **Metode.** Sebanyak lima fitur utama dan dua belas indikator teknikal digunakan untuk membentuk berbagai konfigurasi input yang diterapkan pada data saham Google, Apple, dan Microsoft. **Hasil.** Evaluasi menunjukkan bahwa model berbasis input paralel seperti LSTM-P dan 2LSTM-P mampu meningkatkan akurasi prediksi hingga 8–12% dibandingkan pendekatan stacked konvensional, sekaligus menurunkan bias prediksi lebih dari 50% dalam beberapa kasus. **Kesimpulan.** Arsitektur paralel terbukti efektif dalam mengelola kompleksitas fitur multivariabel, mengurangi *overfitting*, serta mempertahankan konsistensi prediksi terhadap fluktuasi harga. Pendekatan ini juga memungkinkan pemisahan jalur representasi fitur yang lebih terstruktur, sehingga memperkuat kemampuan model dalam mempelajari dinamika temporal jangka panjang. Penelitian ini memberikan kontribusi konseptual dan empiris dalam pengembangan baseline prediksi harga saham berbasis deep learning yang sistematis, sekaligus membuka arah baru untuk eksplorasi tuning adaptif, *attention mechanism*, dan integrasi model hybrid pada sistem keuangan cerdas..

Keywords: Arsitektur Input Paralel; Deret Waktu Multivariat; Evaluasi Kinerja Prediktif; LSTM Bertingkat; Prediksi Harga Saham; Pembelajaran Mendalam; Rekayasa Fitur.

2.2 Pendahuluan

Pasar saham merupakan salah satu elemen penting dalam sistem keuangan global yang memiliki karakteristik dinamis, kompleks, dan penuh ketidakpastian. Pergerakan harga saham yang volatil menyebabkan kebutuhan akan model prediktif yang akurat semakin meningkat, khususnya untuk mendukung pengambilan keputusan investasi yang berbasis data. Dalam beberapa tahun terakhir, pendekatan

buatan, terutama *deep learning*, telah menunjukkan potensi menganalisis pola historis dan memproyeksikan harga saham al., 2024; Karim et al., 2023). Berbagai pendekatan tersebut, *Memory* (LSTM) menonjol karena kemampuannya dalam ntungan jangka panjang dalam data deret (Hochreiter & Sebastian & Tantia, 2025; Wen & Li, 2023).

Sebagian besar penelitian yang menggunakan LSTM untuk prediksi harga saham cenderung fokus pada teknik tuning parameter dan regularisasi untuk meningkatkan performa model (Bhuiyan et al., 2025; Gupta, Kumar Gupta, et al., 2023; U. Singh et al., 2024). Walaupun strategi ini efektif, pendekatan tersebut sering kali menyembunyikan performa dasar (baseline) alami dari arsitektur model. Padahal, dalam konteks penelitian ilmiah, evaluasi performa awal dari sebuah model tanpa intervensi parameter sangat penting untuk memahami batas kemampuan strukturalnya. Dengan kata lain, eksplorasi model *stacked*-LSTM multivariat tanpa tuning dan regularisasi dapat memberikan wawasan awal mengenai seberapa efektif struktur model dalam mempelajari dinamika data finansial secara murni (Shao, 2024; Tuan et al., 2022; Y. Xu et al., 2020).

Penelitian ini bertujuan untuk menyusun dan mengevaluasi arsitektur *stacked*-LSTM multivariabel dengan menggabungkan fitur utama (seperti *open, close, high, low, volume*) dan fitur rekayasa teknikal (seperti *Moving Average*, *RSI, MACD*, dan lain-lain) tanpa menerapkan tuning hyperparameter maupun metode regularisasi (Long et al., 2019). Dengan demikian, penelitian ini menawarkan analisis eksploratif terhadap performa asli model deep learning berbasis LSTM dalam menangani data saham multivariabel. Model yang dibangun bersifat statis dan tidak disesuaikan melalui optimisasi parameter, sehingga hasil yang diperoleh mencerminkan performa dasar yang dapat digunakan sebagai titik awal penelitian lanjutan (Noh et al., 2023; Zhou et al., 2022).

Kontribusi utama dari studi ini terletak pada penyediaan baseline model prediksi saham berbasis *stacked*-LSTM multivariabel yang disusun secara sistematis tanpa intervensi tuning maupun regulasi, selain itu, mekanisme input variable menjadi salah satu bagian penting yang dijelaskan dalam studi ini. Penelitian ini diharapkan dapat memberikan dasar perbandingan yang objektif bagi studi-studi berikutnya yang hendak mengeksplorasi teknik tuning, ensemble learning, dan integrasi mekanisme attention. Selain itu, hasil penelitian ini juga memberikan wawasan tentang bagaimana pengaruh kombinasi fitur utama dan fitur rekayasa terhadap performa prediktif model *deep learning* dalam domain keuangan (Aseeri, 2023; Dwiandiyanta et al., 2025; Moodi et al., 2024).

2.3 Metode Penelitian

2.3.1 Desain Penelitian

Penelitian ini menggunakan desain eksperimen komputasi kuantitatif dengan pendekatan multivariabel time-series *forecasting* berbasis *stacked*-LSTM. Fokus penelitian adalah memprediksi harga saham dengan memanfaatkan rekayasa fitur

ham perusahaan teknologi besar.

juga menerapkan strategi multivariabel forecasting, di mana digunakan secara bersamaan untuk memprediksi target harga horizon waktu tertentu. Dengan demikian, desain penelitian ini

a. Mengumpulkan data time-series multivariabel saham Google (GOOGL), Apple (AAPL), dan Microsoft (MSFT) pada periode 1 Oktober 2004 hingga 30 Desember 2024.
b. Membangun fitur teknikal (*feature engineering*) yang relevan dan informatif bagi pola harga saham.
c. Mengembangkan arsitektur *Stacked*-LSTM dengan struktur bertingkat untuk menangkap dependensi temporal jangka pendek dan panjang dengan mekanisme input yang bervarisi berdasarkan jenis fitur yaitu input tunggal dan parallel serta melakukan penyetelan fitur menjadi input secara bersama-sama atau parsial.
d. Melatih dan mengevaluasi model menggunakan metrik yang dipilih untuk mengukur keakuratan dan bias prediksi model.
e. Menginterpretasi hasil prediksi untuk memastikan model memiliki generalisasi yang baik pada data uji.

2.3.2 Data dan Sumber Data

Penelitian ini menggunakan data historis time-series multivariabel dari tiga saham perusahaan teknologi besar, yaitu Google (GOOGL), Apple (AAPL), dan Microsoft (MSFT), dengan periode 1 Oktober 2004 hingga 30 Desember 2024. Data diambil dari Yahoo Finance (https://finance.yahoo.com) yang menyediakan informasi harga saham secara komprehensif dan telah banyak digunakan dalam penelitian prediksi harga saham berbasis machine learning dan deep learning.

Fitur utama yang digunakan meliputi: *Open* (harga pembukaan), *High* (harga tertinggi harian), *Low* (harga terendah harian), *Close* (harga penutupan), *Volume* (volume transaksi harian). Selain itu, penelitian ini menambahkan fitur teknikal hasil rekayasa (feature engineering) dari kelima variabel tersebut untuk meningkatkan akurasi prediksi model.

2.3.3 Pra-pemrosesan Data

Tahap pra-pemrosesan data dilakukan untuk memastikan kualitas dan kesiapan data sebelum masuk ke proses rekayasa fitur dan pemodelan. Langkah-langkah utama meliputi:

a. Penanganan missing values dengan imputasi median atau interpolasi linier.
b. Normalisasi data menggunakan Min-Max Scaling [0,1] (Rataj et al., 2023), untuk stabilitas model.
c. Pembentukan window time-series menjadi input sequences dan target untuk pelatihan *Stacked*-LSTM. Window time-series ditetapkan 40 input data

(Feature Engineering)

ur dilakukan untuk menambah informasi teknikal yang relevan saham. Fitur yang dihitung meliputi:

- *Moving Average (MA)*
- *Exponential Moving Average (EMA)*
- *Relative Strength Index (RSI)*
- *Moving Average Convergence Divergence (MACD)*
- *Volume Weighted Average Price (VWAP)*
- *High-Low Range*
- *Average True Range (ATR)*
- *On Balance Volume (OBV)*
- *Bollinger Bands (Upper & Lower)*
- *Stochastic Oscillator (%K dan %D)*

Perhitungan dilakukan untuk masing-masing saham (Google, Apple, dan Microsoft) menggunakan data harga dan volume historis.

2.3.5 Pemisahan Dataset

Dataset dibagi menjadi dua bagian yaitu data latih (*training set*): 80% dari data total, digunakan untuk melatih model. Data uji (*testing set*): 20% dari data total, digunakan untuk mengevaluasi kinerja model. Pemisahan dilakukan tanpa *reshuffling* untuk menjaga urutan temporal data time-series.

2.3.6 Pengembangan Model

Model yang digunakan adalah *Stacked*-LSTM dengan arsitektur dua hingga tiga lapisan LSTM bertingkat untuk menangkap pola temporal jangka pendek dan panjang. Konfigurasi model menggunakan fungsi aktivasi tanh dengan optimizer Adam. Untuk mengukur model selama proses training digunakan fungsi loss *Mean Squared Error* (MSE). Output model berupa Dense layer untuk memprediksi harga penutupan saham pada horizon waktu berikutnya.

2.3.7. Pelatihan Model

Model dilatih dengan konfigurasi sebagai berikut:

a. Unit pada layer
   - LSTM = 100
   - 2LSTM = Unit layer-1 = 150, Unit layer-2 = 100
   - 3LSTM = Unit layer-1 = 150, Unit layer-2 = 150, Unit layer-3 = 100,
b. Epoch = 50
c. Batch size: 32
d. Learning rate = 0.001 (Default)

olit: 20% dari data latih digunakan untuk validasi selama

ja Model

nggunakan metrik regresi berikut:

quare Error (RMSE) (Hyndman & Athanasopoulos, 2018)

b. *Mean Absolute Percentage Error* (MAPE) (Makridakis et al., 2018)
c. *Root Mean Square Percentage Error* (RMSPE) (Lawi et al., 2022)
d. *R-Squared* (R^2) (John & Latha, 2023)
e. Percent Bias (PBIAS) (Samantaray et al., 2025)

2.3.9 Tools dan Implementasi

Bahasa pemrograman: Python
Library: TensorFlow/Keras, Pandas, NumPy, Scikit-learn, Matplotlib, TA-Lib (untuk fitur teknikal).

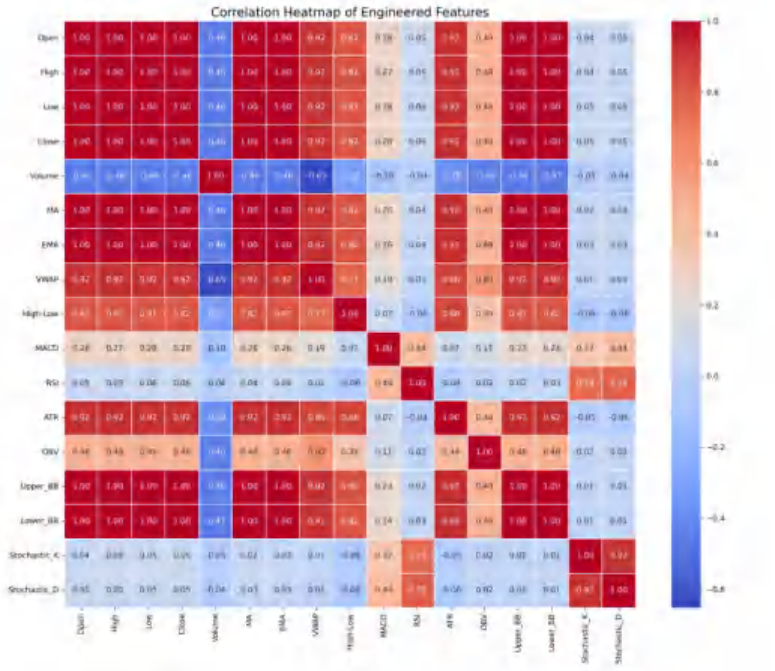
2.4 Hasil dan Pembasan

2.4.1 Hasil Penelitian

Bagian ini menyajikan hasil penelitian yang diperoleh dari implementasi model *Stacked*-LSTM dengan rekayasa fitur teknikal pada data saham Google, Apple, dan Microsoft. Hasil yang ditampilkan mencakup evaluasi kinerja model berdasarkan metrik regresi, visualisasi prediksi dibandingkan data aktual, serta analisis singkat terhadap performa model pada dataset uji. Hasil yang disajikan dilengkapi dengan motode pembelajaran mesin dan *neural network* lainnya sebagai pembanding dalam penelitian ini.

a. Korelasi Pearson antar fitur pada Saham Google

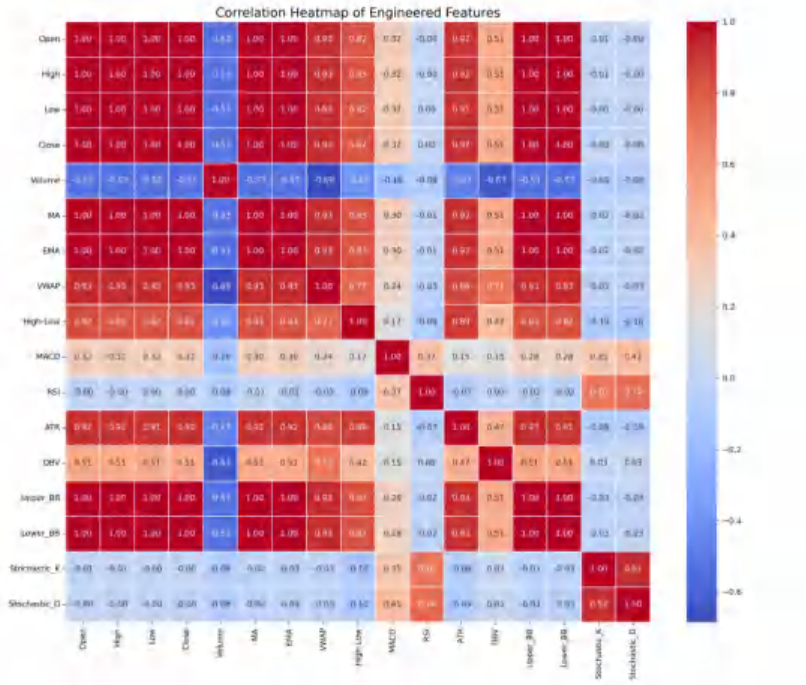
Heatmap korelasi yang disajikan pada Gambar 2.1 menunjukkan hubungan linear antara fitur utama dan fitur rekayasa yang digunakan dalam penelitian ini. Terlihat bahwa fitur utama seperti Open, High, Low, dan Close memiliki korelasi positif yang sangat tinggi ($r \approx 1.00$), yang wajar mengingat keterkaitan struktural antar harga dalam data saham harian. Fitur rekayasa seperti *Moving Average (MA), Exponential Moving Average (EMA), Volume Weighted Average Price (VWAP), Average True Range (ATR), On Balance Volume (OBV),* dan *Upper Bollinger Band (Upper_BB)* juga menunjukkan korelasi positif tinggi terhadap harga penutupan, yang mengindikasikan relevansinya dalam menangkap informasi tren harga. Sebaliknya, Volume menunjukkan korelasi negatif sedang (-0.46) terhadap sebagian besar fitur harga, menunjukkan bahwa variabel ini memuat informasi berbeda terkait aktivitas pasar dibandingkan pergerakan harga itu sendiri. Indikator momentum dan osilator seperti MACD, RSI, Stochastic-K, dan Stochastic-D memperlihatkan korelasi rendah hingga sedang, menegaskan potensinya dalam menyediakan sinyal prediksi yang bersifat komplementer dan tidak redundant. Temuan ini menunjukkan bahwa fitur rekayasa yang digunakan mencakup informasi yang bersifat sangat berkorelasi sehingga memperkaya ruang input untuk pelatihan model deep .



Gambar 2.1 Heatmap Koralasi antar fitur pada Saham Google

b. Korelasi Pearson antar fitur pada Saham Apple

Gambar 2.2 menampilkan heatmap korelasi antara fitur utama dan fitur rekayasa yang digunakan pada data saham Apple, yang menggambarkan kekuatan hubungan linear antar variabel berdasarkan koefisien korelasi Pearson. Terlihat bahwa fitur utama Open, High, Low, dan Close memiliki korelasi sangat tinggi ($r \approx 1.00$), menunjukkan keterkaitan struktural harga harian yang kuat. Fitur rekayasa MA, EMA, VWAP, Upper_BB, dan Lower_BB juga menunjukkan korelasi tinggi dengan harga penutupan, menegaskan perannya dalam menangkap tren harga yang stabil dan fluktuasi pasar. Sebaliknya, Volume memiliki korelasi negatif sedang (-0.53) terhadap variabel harga, menandakan bahwa volume transaksi memuat dimensi informasi yang berbeda terkait intensitas aktivitas pasar. Indikator momentum dan osilator seperti MACD, RSI, serta Stochastic-K dan Stochastic-D menunjukkan korelasi rendah hingga sedang terhadap harga penutupan, yang menunjukkan potensinya dalam memberikan sinyal prediksi komplementer terkait kekuatan tren dan kondisi jenuh beli atau jenuh jual. Analisis ini menegaskan bahwa fitur-fitur rekayasa yang digunakan mencakup informasi tren yang berkorelasi tinggi sekaligus sinyal teknikal ingga memperkaya representasi input model dan mendukung deep learning yang lebih akurat, generalisatif, serta tahan :ting dalam peramalan harga saham Apple.



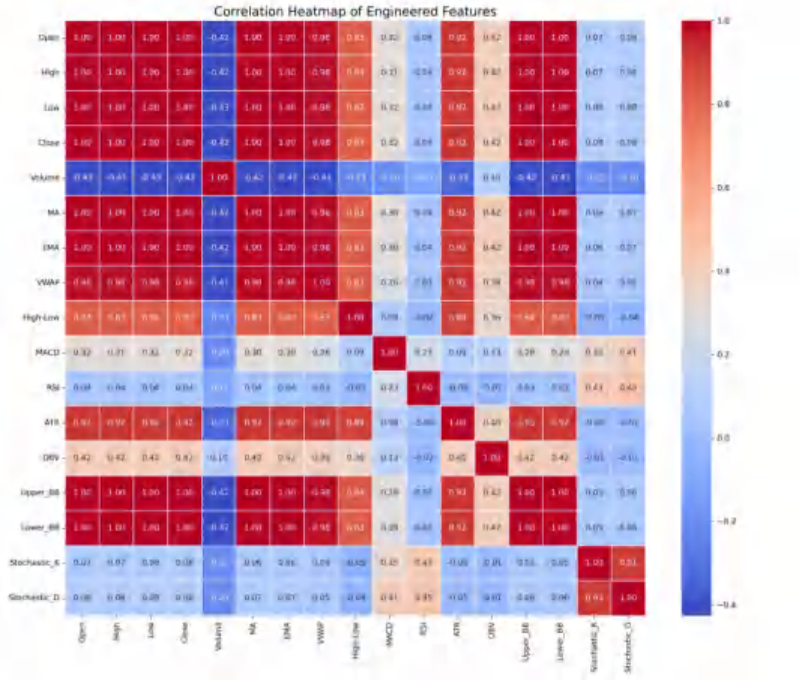
Gambar 2.2 Heatmap Koralasi antar fitur pada Saham Apple

c. Korelasi Pearson antar fitur pada Saham Microsoft

Heatmap korelasi pada Gambar 2.3 memaparkan pola keterkaitan linear antara fitur utama dan fitur rekayasa pada data saham Microsoft. Analisis menunjukkan bahwa variabel harga utama—Open, High, Low, dan Close—memiliki koefisien korelasi mendekati sempurna (r ≈ 1.00), yang mencerminkan relasi struktural inheren dalam dinamika pergerakan harga saham intraday. Fitur rekayasa seperti Moving Average (MA), Exponential Moving Average (EMA), Volume Weighted Average Price (VWAP), Upper Bollinger Band, dan Lower Bollinger Band juga menunjukkan korelasi sangat tinggi dengan harga penutupan, yang menegaskan validitasnya sebagai indikator tren yang merepresentasikan pergerakan harga rata-rata dan batas volatilitas pasar. Sementara itu, fitur Volume memiliki korelasi negatif sedang (-0.42) terhadap variabel harga, menandakan adanya informasi tambahan yang berkaitan dengan intensitas transaksi dan tekanan pasar yang tidak sepenuhnya dijelaskan oleh harga. Di sisi lain, indikator momentum dan osilator seperti MACD, RSI, Stochastic-K, dan Stochastic-D menunjukkan


ngga sedang, yang mengindikasikan kapasitasnya dalam
l non-redundan untuk mendeteksi titik pembalikan tren dan
eli atau jenuh jual. Secara keseluruhan, distribusi korelasi ini
kombinasi fitur utama dan rekayasa yang digunakan dalam
i potensi kuat untuk mendukung pembelajaran model deep
engan struktur input yang informatif dan representatif, sehingga

meningkatkan akurasi dan kemampuan generalisasi prediksi harga saham Microsoft di pasar keuangan yang kompleks dan dinamis.



Gambar 2.3 Heatmap Koralasi antar fitur pada Saham Microsoft

d. Evaluasi Performa Model Berdasarkan Arsitektur dan Konfigurasi Input

Penyajian metrik mencakup nilai akurasi dan error yang diperoleh dari proses pengujian model terhadap data uji, sehingga memungkinkan analisis komparatif atas efektivitas dan kemampuan generalisasi model dalam memprediksi harga saham. Hasil evaluasi ini menjadi dasar penting dalam menentukan model dengan performa terbaik untuk diimplementasikan pada studi kasus pasar saham yang dianalisis.

1) Evaluasi Model Berdasarkan Arsitektur dan Konfigurasi Input Saham *Google*

Hasil evaluasi yang disajikan dalam Tabel 2.1 menunjukkan variasi substansial dalam kinerja prediktif tergantung pada kedalaman arsitektur dan konfigurasi fitur yang digunakan. Dalam konteks perkiraan saham Google, hasilnya menunjukkan bahwa pilihan struktur input baik hanya didasarkan pada fitur utama, ombinasi hibrida, secara signifikan memengaruhi generalisasi rkiraan.

Tabel 2.1 Metrik Evaluasi Model Berdasarkan Arsitektur dan Konfigurasi Input pada Saham Google

Arsitektur	Fitur	RMSE	MAE	MAPE (%)	RMSPE (%)	R2 (%)	PBIAS (%)
MLP	Utama	4.2538	3.3143	2.5	3.11	97.34	1.29
SVR	Utama	5.3609	3.8655	2.62	3.29	95.8	2.88
VanilaRNN	Utama	3.9773	3.2039	2.38	2.87	97.54	2.03
MLP	Rekayasa	7.3741	5.5919	4.21	5.41	91.76	3.93
SVR	Rekayasa	10.5165	7.9059	5.4	6.62	83.83	5.96
VanilaRNN	Rekayasa	6.8261	5.9426	4.38	4.83	92.76	4.41
MLP	Utama + Rekayasa	5.9956	4.5737	3.55	4.62	94.71	-0.42
SVR	Utama + Rekayasa	18.5138	14.3612	9.85	11.66	49.88	11.03
VanilaRNN	Utama + Rekayasa	3.3096	2.4704	1.84	2.38	98.3	0.94
LSTM	Utama	3.2691	2.5667	1.94	2.44	98.34	1.31
2LSTM	Utama	3.0167	2.3351	1.78	2.27	98.59	1.1
3LSTM	Utama	**2.7401**	**2.0145**	**1.58**	**2.16**	**98.83**	**-0.31**
LSTM	Rekayasa	4.1666	3.1939	2.36	2.95	97.3	1.16
2LSTM	Rekayasa	5.9108	4.617	3.35	4.03	94.57	2.34
3LSTM	Rekayasa	15.1739	12.034	8.28	9.69	64.24	8.52
LSTM	Utama + Rekayasa	6.0083	4.7905	3.39	3.98	94.39	3.31
2LSTM	Utama + Rekayasa	6.8325	5.3464	3.76	4.45	92.75	3.52
3LSTM	Utama + Rekayasa	14.5255	11.858	8.22	9.37	67.23	8.81
	Utama + Rekayasa	3.5732	2.7481	2.03	2.54	98.02	1.3
	Utama + Rekayasa	**3.4765**	**2.6232**	**2.08**	**2.8**	**98.12**	**-0.57**
	Utama + Rekayasa	5.0224	3.8349	2.8	3.48	96.08	1.59
STM	Utama + Rekayasa	3.835	2.9174	2.14	2.68	97.72	1.23
LSTM	Utama + Rekayasa	5.3173	3.9478	2.79	3.48	95.61	2.01
LSTM	Utama + Rekayasa	11.3482	9.276	6.45	7.34	80	6.94

Di antara model input tunggal (tanpa struktur paralel atau layer pasca-penggabungan), LSTM 3-layer (3LSTM) dengan hanya fitur utama mencapai kinerja paling kuat, mencatat RMSE 2.7401, MAPE 1.58%, dan R^2 98.83%. Ini menegaskan efektivitas kedalaman temporal dalam menangkap dependensi jangka panjang ketika fitur input secara inheren terstruktur dan sangat berkorelasi, seperti yang dibuktikan sebelumnya dalam heatmap korelasi saham Google.

Namun, ketika model dilatih hanya menggunakan fitur rekayasa, semua arsitektur mengalami penurunan akurasi yang nyata. Misalnya, 3LSTM dengan fitur rekayasa saja menghasilkan RMSE yang jauh lebih tinggi sebesar 15.1739 dan R^2 yang lebih rendah sebesar 64.24%, menunjukkan bahwa fitur yang direkayasa, meskipun informatif, tidak secara independen memberikan representasi temporal yang memadai untuk mendorong prediksi yang tepat. Hasil ini menggarisbawahi pentingnya mengintegrasikan sinyal pasar untuk fitur utama yaitu open, high, low, dan close, dalam menangkap dinamika harga.

Konfigurasi input hibrid yang menggabungkan fitur utama dan rekayasa memberikan hasil yang beragam. Menariknya, arsitektur Vanilla RNN dengan input gabungan mencapai MAPE terendah (1.84%) di antara semua model dan skor R^2 yang kuat sebesar 98.30%, menyoroti potensi struktur RNN yang lebih sederhana untuk berkinerja kompetitif di bawah gabungan fitur.

Berlanjut ke arsitektur input paralel (dilambangkan dengan simbol "P"), hasil pengamatan menunjukkan peningkatan yang lebih konsisten. Model 2LSTM-P menggunakan konfigurasi input gabungan mencatat RMSE 3.4765, MAPE 2.08%, dan R^2 98.12%, yang mencerminkan keseimbangan yang menguntungkan antara kompleksitas arsitektur dan pemanfaatan fitur. Kemampuan struktur paralel untuk memproses aliran fitur individu secara independen sebelum fusi kemungkinan berkontribusi pada peningkatan ekstraksi sinyal dan pengurangan kesalahan.

Namun demikian, memperkenalkan LSTM tambahan setelah penggabungan (model dengan PC-LSTM) menghasilkan pengembalian yang berkurang di luar kedalaman tertentu. Sementara LSTM-PC-LSTM masih berkinerja andal (MAPE 2.14%, R^2 97.72%), versi yang lebih dalam seperti 2LSTM-PC-LSTM dan 3LSTM-PC-LSTM mengalami penurunan kinerja (R^2 turun masing-masing menjadi 95.61% dan 80.00%), kemungkinan karena overfitting atau redundansi arsitektur yang berlebihan.

Melengkapi metrik evaluasi sebelumnya, PBIAS berfungsi sebagai indikator penting untuk mengukur arah kesalahan prediksi, apakah model cenderung overestimasi (positif) atau underestimasi (negatif) harga saham sebenarnya. Pada Tabel 2.1, model dengan fitur rekayasa saja menunjukkan nilai PBIAS positif tinggi (misalnya 8.52% pada 3LSTM, 5.96% pada SVR), menandakan kecenderungan

ilangnya sinyal pasar dengan fitur utama. Sebaliknya, model
aik tunggal maupun gabungan, memiliki nilai PBIAS mendekati
nya -0.31% pada 3LSTM, -0.57% pada 2LSTM-P), menunjukkan
ıg dan minim bias. Ini menegaskan bahwa PBIAS penting untuk
:urasi lain dengan memberi gambaran tentang keseimbangan
konfigurasi model.

Tabel 2.2 Metrik Evaluasi Model Berdasarkan Arsitektur dan Konfigurasi Input pada Saham Apple

Arsitektur	Fitur	RMSE	MAE	MAPE (%)	RMSPE (%)	R2 (%)	PBIAS (%)
MLP	Utama	12.4355	11.7464	7.28	7.8	84.37	-7.05
SVR	Utama	8.5188	6.1689	3.28	4.07	97.7	3.67
VanilaRNN	Utama	4.2051	3.4415	2.14	2.66	98.18	-1.73
MLP	Rekayasa	5.2813	4.2367	2.6	3.25	97.07	0.25
SVR	Rekayasa	10.2291	7.4284	4.05	5.13	89.46	4.35
VanilaRNN	Rekayasa	4.3276	3.4939	2.11	2.6	98.07	1.61
MLP	Utama + Rekayasa	6.1218	4.7882	2.89	3.66	96.21	1.14
SVR	Utama + Rekayasa	16.065	11.482	6.08	7.69	74.01	6.89
VanilaRNN	Utama + Rekayasa	12.4172	11.5507	6.78	7.09	84.15	6.88
LSTM	Utama	**3.8321**	**3.0122**	**1.76**	**2.18**	**98.49**	**1.15**
2LSTM	Utama	4.7967	3.7899	2.17	2.62	97.64	1.77
3LSTM	Utama	7.5288	6.2152	3.5	4	94.17	3.52
LSTM	Rekayasa	5.4141	4.1157	2.36	2.97	97	1.57
2LSTM	Rekayasa	8.0181	6.0054	3.31	4.09	93.39	3.07
3LSTM	Rekayasa	30.4554	26.6277	14.89	15.86	4.67	15.85
LSTM	Utama + Rekayasa	18.01	16.04	9.1	9.61	66.65	9.55
2LSTM	Utama + Rekayasa	9.3661	6.3434	3.39	4.53	90.98	2.48
3LSTM	Utama + Rekayasa	24.3007	20.5589	11.37	12.44	39.31	12.24
	Utama + Rekayasa	**3.6191**	**2.8052**	**1.72**	**2.24**	**98.65**	**-0.45**
	Utama + Rekayasa	4.1965	3.2599	1.99	2.59	98.19	-0.48
	Utama + Rekayasa	4.826	3.9341	2.38	2.95	97.61	-2.03
_STM	Utama + Rekayasa	6.8517	5.2693	2.9	3.49	95.17	2.82
-LSTM	Utama + Rekayasa	17.8653	15.6295	8.77	9.37	67.19	9.3
-LSTM	Utama + Rekayasa	25.2514	22.3227	12.55	13.26	34.46	13.29

Hasil optimal prediksi saham Google, dapat diperoleh melalui arsitektur yang seimbang dengan cermat yang menggabungkan kedalaman temporal dengan perlakuan fitur modular. Temuan dalam Tabel 2.1 mengungkapkan bahwa struktur paralel tingkat menengah (misalnya, 2LSTM-P) adalah yang paling efisien dalam memanfaatkan manfaat fitur primer dan rekayasa tanpa menimbulkan penalti kedalaman atau kompleksitas yang tidak perlu.

2) Evaluasi Model Berdasarkan Arsitektur dan Konfigurasi Input Saham *Apple*

Hasil evaluasi model prediksi harga saham Apple sebagaimana tercantum dalam Tabel 2.2 menunjukkan bahwa akurasi model sangat bergantung pada jenis input dan arsitektur yang digunakan. Berdasarkan metrik utama seperti RMSE, MAPE, RMSPE, dan R^2, model dengan input fitur utama cenderung menghasilkan performa yang lebih stabil dan akurat dibandingkan dengan input berupa fitur rekayasa atau kombinasi keduanya. Seperti pada Vanilla RNN dengan input fitur utama mencatat nilai RMSE sebesar 4.2051, MAPE 2.14%, dan R^2 98.18%, menegaskan efektivitas model sederhana dalam menangkap pola deret waktu saat fitur input original dan informatif. LSTM 1-layer juga menunjukkan performa unggul (RMSE 3.8321, MAPE 1.76%, R^2 98.49%) dengan input fitur utama, membuktikan kemampuan arsitektur ini dalam mengelola dependensi temporal.

Penggunaan fitur rekayasa secara eksklusif cenderung menurunkan performa pada sebagian besar arsitektur, terutama saat digunakan bersama model LSTM berlapis. Misalnya, 3LSTM dengan input fitur rekayasa menghasilkan RMSE sebesar 30.4554, MAPE 14.89%, RMSPE 15.86%, dan R^2 hanya 4.67%, yang menunjukkan terjadinya overfitting parah serta kegagalan model dalam menyesuaikan kompleksitas arsitektur dengan representasi fitur. Sementara itu, kombinasi fitur utama dan rekayasa tidak serta-merta meningkatkan akurasi. Model seperti 3LSTM dengan input gabungan justru menunjukkan kinerja buruk (RMSE 24.3007, MAPE 11.37%, R^2 39.31%), menunjukkan bahwa penambahan fitur tanpa strategi segmentasi yang tepat dapat menyebabkan noise dan redudansi yang merugikan proses pembelajaran.

Arsitektur dengan input paralel seperti LSTM-P dan 2LSTM-P memberikan performa paling konsisten dan unggul. LSTM-P mencatat RMSE 3.6191, MAPE 1.72%, RMSPE 2.24%, dan R^2 98.65%, sedangkan 2LSTM-P memperlihatkan RMSE 4.1965, MAPE 1.99%, dan R² 98.19%. Kedua model ini juga memiliki RMSPE rendah dan menunjukkan bahwa pemisahan jalur pemrosesan fitur sebelum penggabungan mampu mengurangi redundansi dan meningkatkan efisiensi representasi. Sebaliknya, arsitektur PC-LSTM yang menambahkan lapisan LSTM

ın menunjukkan kinerja menurun secara signifikan seiring
ıan, sebagaimana terlihat pada 3LSTM-PC-LSTM yang
tinggi (25.2514) dan R^2 rendah (34.46%).

Tabel 2.3 Metrik Evaluasi Model Berdasarkan Arsitektur dan Konfigurasi Input pada Saham Microsoft

Arsitektur	Fitur	RMSE	MAE	MAPE (%)	RMSPE (%)	R2 (%)	PBIAS (%)
MLP	Utama	9.3945	7.4707	2.53	3.32	98.22	-0.94
SVR	Utama	23.8263	17.2591	4.71	5.96	88.59	5.5
VanilaRNN	Utama	8.8493	6.9074	2.06	2.52	98.36	1.54
MLP	Rekayasa	16.2685	12.3061	3.67	4.61	94.59	3.09
SVR	Rekayasa	24.7727	16.9802	4.66	6.2	87.65	4.28
VanilaRNN	Rekayasa	6.707	5.3047	1.74	2.22	99.06	0.43
MLP	Utama + Rekayasa	19.5642	15.3622	4.62	5.6	92.26	3.86
SVR	Utama + Rekayasa	42.8907	29.2509	7.8	10.56	62.98	9.09
VanilaRNN	Utama + Rekayasa	17.5038	14.2544	4.15	4.75	93.58	4.31
LSTM	Utama	6.6342	5.1372	1.66	2.15	99.08	-1.17
2LSTM	Utama	7.4313	6.2022	1.98	2.36	98.84	1.56
3LSTM	Utama	**6.465**	**5.0816**	**1.68**	**2.2**	**99.12**	**-0.34**
LSTM	Rekayasa	19.3644	16.4969	4.88	5.4	92.14	5.11
2LSTM	Rekayasa	38.0429	31.55	9.1	10.1	69.66	9.87
3LSTM	Rekayasa	52.9742	43.0077	12.21	13.88	41.17	13.52
LSTM	Utama + Rekayasa	20.3886	16.9415	4.92	5.52	91.29	5.23
2LSTM	Utama + Rekayasa	46.2608	39.7089	11.58	12.49	55.14	12.53
3LSTM	Utama + Rekayasa	61.8176	52.1319	15.05	16.44	19.89	16.45
	Utama + Rekayasa	**6.3058**	**4.9103**	**1.57**	**2.03**	**99.17**	**0.58**
	Utama + Rekayasa	8.6323	6.6838	2.2	2.9	98.44	-1.6
	Utama + Rekayasa	9.1728	7.1751	2.29	2.95	98.24	0.26
STM	Utama + Rekayasa	16.307	12.9364	3.71	4.32	94.42	3.83
LSTM	Utama + Rekayasa	17.4229	13.2132	3.72	4.51	93.63	3.69
LSTM	Utama + Rekayasa	35.1437	28.0896	7.93	9.14	74.11	8.79

Keandalan evaluasi ini diperkuat oleh metrik PBIAS, yang mengukur kecenderungan arah prediksi. Model-model dengan akurasi tinggi umumnya menunjukkan nilai PBIAS yang mendekati nol, seperti LSTM-P (–0.45%) dan 2LSTM-P (–0.48%), yang menunjukkan prediksi yang seimbang tanpa kecenderungan sistematis untuk tidak over atau under tarhadap harga saham. Sebaliknya, model dengan performa buruk cenderung memiliki PBIAS ekstrem, seperti 3LSTM-PC-LSTM (13.29%) atau 3LSTM dengan fitur rekayasa (15.85%), mengindikasikan bias overestimasi yang kuat dan tidak terkontrol.

3) Evaluasi Model Berdasarkan Arsitektur dan Konfigurasi Input Saham *Microsoft*

Dalam meramalkan harga saham Microsoft, performa model sangat dipengaruhi oleh sejauh mana arsitektur yang digunakan mampu menyeimbangkan akurasi numerik, sensitivitas terhadap jenis input, dan kestabilan arah prediksi. Berdasarkan hasil evaluasi yang disajikan pada Tabel 2.3, terlihat bahwa model-model berbeda menunjukkan variasi kinerja yang signifikan ketika diuji dengan konfigurasi input yang bervariasi dan kedalaman arsitektur yang meningkat. Evaluasi ini menggunakan metrik utama seperti RMSE, MAPE, RMSPE, dan R^2, serta diperkuat oleh analisis nilai PBIAS untuk mengukur arah sistematis dari kesalahan prediksi.

Secara umum, model dengan input fitur utama saja memperlihatkan hasil yang paling solid. LSTM satu lapis mencatat RMSE rendah (6.6342), MAPE hanya 1.66%, dan R^2 sangat tinggi (99.08%), mencerminkan keunggulan arsitektur berulang dalam memanfaatkan sinyal mentah yang bersifat temporal. Vanilla RNN juga mencatat hasil sangat baik dengan fitur utama (MAPE 2.06%, R^2 98.36%), meskipun tanpa keunggulan kompleksitas. Sebaliknya, model non-sekuensial seperti SVR dan MLP menunjukkan performa lebih lemah, terutama saat dipasangkan dengan input gabungan. Hal ini menandakan bahwa fitur utama merupakan sumber informasi yang paling konsisten dan berdampak langsung terhadap akurasi prediksi untuk saham Microsoft. Selain itu, penggunaan fitur rekayasa secara eksklusif sering kali memperburuk akurasi model, terutama pada arsitektur LSTM bertingkat. Seperti, 3LSTM dengan fitur rekayasa mencatat MAPE sangat tinggi (12.21%), RMSPE 13.88%, dan R^2 hanya 41.17%. Ini menunjukkan bahwa meskipun fitur rekayasa membawa nilai tambah dalam banyak kasus namun tidak dapat menggantikan fungsi sinyal primer dari data harga. Bahkan dengan model yang sangat kuat secara teoritis, fitur rekayasa tanpa dukungan harga aktual dapat memperbesar kesalahan prediksi. Ketika fitur utama dan rekayasa digabungkan, hasilnya sangat tergantung pada

alam model standar seperti 3LSTM atau SVR, gabungan ini kompleksitas tanpa menambah nilai informatif, seperti terlihat (RMSE 42.8907, R^2 62.98%) atau 3LSTM gabungan (MAPE .89%). Hal ini menegaskan pentingnya pengelolaan jalur input dimasukkan bersifat heterogen.

ut paralel (LSTM-P) menawarkan solusi atas tantangan ini. s fitur utama dan rekayasa secara terpisah sebelum

penggabungan, model seperti LSTM-P dan 2LSTM-P terbukti jauh lebih efisien dan akurat. LSTM-P, misalnya, mencatat RMSE hanya 6.3058, MAPE 1.57%, dan R^2 99.17%, dengan RMSPE rendah (2.03%), menjadikannya arsitektur terbaik untuk saham Microsoft. Bahkan ketika lapisan ditambah menjadi 2LSTM-P dan 3LSTM-P, performa tetap kompetitif, menunjukkan bahwa struktur modularisasi input memperkuat kestabilan model terhadap peningkatan kompleksitas.

Pentingnya arah kesalahan prediksi tercermin dari nilai PBIAS. Model dengan akurasi tinggi seperti LSTM-P (PBIAS 0.58%) dan 2LSTM-P (–1.60%) memperlihatkan prediksi yang seimbang, tanpa kecenderungan sistematis untuk over- atau under-estimate. Sebaliknya, model dengan performa buruk seperti 3LSTM dengan gabungan input (PBIAS 16.45%) dan SVR gabungan (PBIAS 9.09%) menunjukkan bias overestimasi yang kuat dan membahayakan dalam konteks pengambilan keputusan berbasis harga.

Dengan demikian, performa terbaik dalam prediksi harga saham Microsoft dicapai oleh model dengan arsitektur yang modular, dalam tingkat wajar, dan mengadopsi struktur input paralel, yang tidak hanya akurat tetapi juga bebas dari bias sistematis.

2.4.2 Pembahasan

Bagian ini mendiskusikan implikasi dari hasil analisis korelasi fitur, nilai metrik dari masing-masing model, serta hasil visualisasi prediksi dari model yang telah diperoleh, serta bagaimana temuan tersebut mendukung perancangan arsitektur model dalam penelitian ini. Diskusi difokuskan pada interpretasi hubungan antar fitur utama dan fitur rekayasa, relevansinya terhadap kualitas prediksi harga saham, serta kontribusinya dalam meningkatkan akurasi dan generalisasi model deep learning multivariat yang dikembangkan.

a. Analisis Korelasi Fitur

Analisis korelasi Pearson yang dilakukan terhadap seluruh variabel masukan pada ketiga saham (Google, Apple, dan Microsoft) mengungkapkan adanya pola hubungan yang khas antara fitur harga utama dan fitur teknikal hasil rekayasa. Variabel seperti Open, High, Low, dan Close menunjukkan hubungan yang sangat kuat satu sama lain, dengan koefisien korelasi yang sangat tinggi ($r > 0.95$). Sebaliknya, fitur-fitur teknikal seperti MACD, RSI, ATR, Stochastic Oscillator, dan OBV menunjukkan korelasi yang lebih rendah terhadap fitur utama, dengan nilai Pearson berkisar antara 0.2 hingga 0.6. Hal ini mengindikasikan bahwa fitur teknikal membawa informasi tambahan yang bersifat orthogonal terhadap data harga, dan
aya struktur data multivariat. Dalam kerangka deep learning,
ni sangat penting karena dapat mendorong model untuk belajar
ih dalam dan komprehensif terhadap dinamika pasar keuangan

i temuan korelasi ini sangat signifikan dalam perancangan
diktif. Korelasi tinggi antar harga utama dapat dimanfaatkan

untuk pemodelan sekuensial yang stabil, sementara korelasi rendah antara fitur teknikal dan harga utama justru membuka peluang untuk membangun jalur pemrosesan paralel yang terpisah. Pendekatan ini selaras dengan prinsip-prinsip representasi diferensial dalam pembelajaran mesin, di mana jalur input yang tidak saling tergantung cenderung memperkuat kapasitas generalisasi model (John & Latha, 2023a). Selain itu, Fjellstrom (2022) menyatakan bahwa pemisahan sinyal berdasarkan kategori korelasinya dapat meningkatkan ketahanan model terhadap fluktuasi jangka pendek. Oleh karena itu, pemetaan korelasi tidak hanya menjadi alat eksploratif awal, tetapi juga menjadi dasar yang strategis dalam membangun arsitektur prediktif yang efisien, modular, dan adaptif terhadap struktur data saham yang kompleks. Namun dalam penelitian tidak melakukan pemisahan input berdasarkan korelasi antar fitur.

b. Analisis evaluasi model berdasarkan Metrik Evaluasi Model Setiap Saham

Saham Google menunjukkan performa prediktif terbaik saat menggunakan arsitektur LSTM paralel, khususnya pada konfigurasi 2LSTM-P dengan input kombinasi fitur utama dan rekayasa. Hal ini tercermin dari nilai RMSE dan MAPE yang lebih rendah dibandingkan arsitektur tradisional seperti MLP dan SVR. Namun dengan arsitektur *single input* model 3LSTM yang memiliki keakuratan lebih tinggi dibandingkan dengan model lainnya dengan dengan capaian RMSE sebesar 2.7401, R^2 sebesar 98.83%, serta MAPE 1.58% dan RMSPE 2.16%. Nilai PBIAS –0.31% pada konfigurasi input dengan fitur utama. Selain itu, model VanilaRNN juga memberikan hasil yang kompetitif pada input fitur utama saja, namun tetap kalah stabil dibandingkan LSTM yang memiliki kemampuan mempertahankan konteks jangka panjang. Keunggulan LSTM dalam menangkap dinamika pasar saham jangka menengah telah didukung oleh temuan (Saberironaghi et al., 2025), yang menyatakan bahwa model deep learning berbasis memori seperti LSTM dapat menghasilkan prediksi yang lebih presisi pada data keuangan multivariat berfluktuasi tinggi.

Kemudian saham Apple juga mencerminkan pola yang serupa, di mana model LSTM dan varian paralelnya berhasil mengungguli pendekatan klasik dalam sebagian besar metrik evaluasi. Model LSTM-P menunjukkan kinerja yang sangat baik dengan error prediksi yang rendah dan nilai R^2 > 98. Kinerja optimal ini dapat dikaitkan dengan struktur data saham Apple yang memiliki kombinasi korelasi kuat pada fitur utama dan variabilitas tinggi pada fitur teknikal. Dalam konteks ini, pendekatan modular atau paralel seperti yang diadopsi dalam LSTM-P terbukti

sasi model

aluasi performa model stacked-LSTM pada saham Microsoft pendekatan single input berbasis fitur utama paling optimal am arsitektur 3LSTM, dengan capaian RMSE sebesar 6.4650, serta MAPE 1.68% dan RMSPE 2.20%. Nilai PBIAS –0.34% nya bias prediksi, memperkuat efisiensi model dalam mporal tanpa gangguan fitur tambahan. Hal ini menegaskan

bahwa peningkatan kedalaman LSTM masih menguntungkan selama input dijaga tetap sederhana dan informatif.

Sebaliknya, pada model-model yang mengintegrasikan mekanisme input paralel seperti pada arsitektur LSTM-P dan LSTM-PC-LSTM, performa prediktif juga menunjukkan hasil yang sangat kompetitif dan bahkan dalam beberapa kasus melampaui konfigurasi stacked konvensional. Contohnya, model LSTM-P pada input gabungan mencatat RMSE = 6.3058, R^2 = 99.17%, dan PBIAS = 0.58%, menunjukkan bahwa pemisahan jalur input utama dan fitur rekayasa memberikan kontribusi nyata dalam menjaga akurasi dan mengurangi noise antar fitur. Pendekatan modular ini memungkinkan model untuk mengatur bobot dan konteks temporal secara terpisah terhadap setiap grup fitur, yang pada akhirnya meningkatkan kapasitas generalisasi tanpa kehilangan spesifisitas informasi.

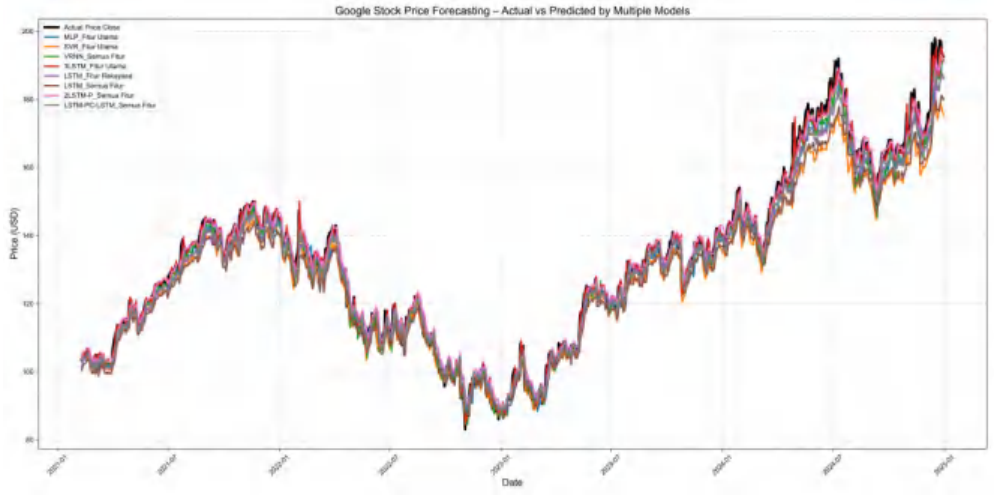
Secara keseluruhan, temuan dari ketiga saham menunjukkan bahwa arsitektur paralel berbasis LSTM, khususnya LSTM-P dan 2LSTM-P, mampu memberikan performa prediksi yang paling konsisten dan akurat pada konfigurasi input gabungan. Arsitektur LSTM dengan konfigurasi *single input* untuk fitur utama yang menunjukkan kestabilan metrik yaitu model stacked-LSTM dengan 3-Layer (3LSTM) baik pada saham Google maupun saham Microsoft. Untuk basis LSTM memiliki performa baik pada konfigurasi single input baik pada fitur utama maupun fitur rekayasa. Keunggulan ini terutama terlihat pada kestabilan metrik error seperti RMSE, MAPE, dan RMSPE, serta pencapaian nilai koefisien determinasi (R^2) yang tinggi dan nilai PBIAS yang mendekati nol, yang menandakan keseimbangan antara akurasi dan bias estimasi. Model-model konvensional seperti MLP dan SVR menunjukkan performa yang bervariasi tergantung jenis fitur yang digunakan, dan cenderung sensitif terhadap penambahan fitur rekayasa yang kompleks. Sebaliknya, pendekatan deep learning dengan desain jalur pemrosesan paralel terbukti lebih adaptif terhadap karakteristik data multivariat dan mampu menangani redundansi maupun konflik antar fitur. Hal ini memperkuat argumen bahwa modularitas dan desain arsitektur berbasis informasi korelasi dapat secara signifikan meningkatkan kemampuan generalisasi model prediktif dalam domain pasar saham yang sangat dinamis

c. Analisis visualisasi harga saham aktual dengan hasil prediksi multi model

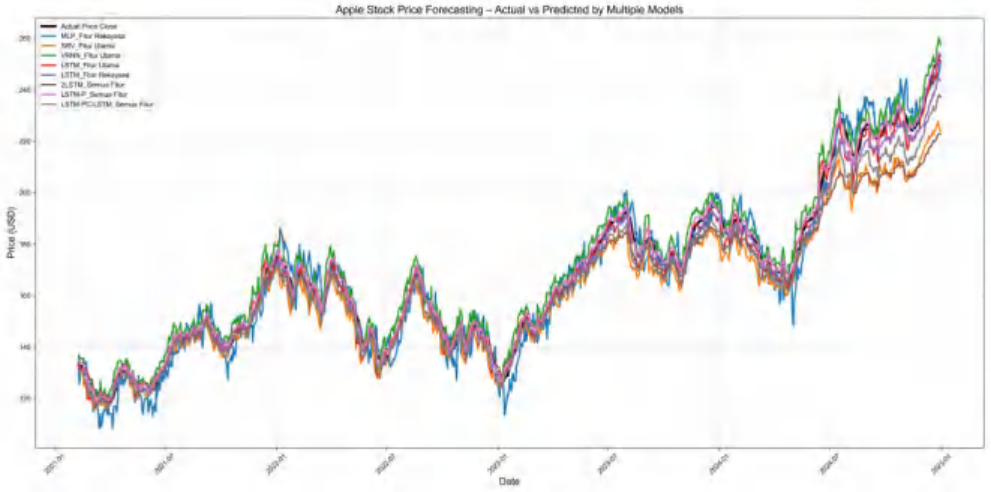
Visualisasi komparatif antara harga saham aktual dan hasil prediksi dari berbagai model menjadi instrumen penting dalam mengevaluasi kinerja model prediktif secara lebih intuitif dan interpretatif. Representasi grafis ini tidak hanya membantu dalam memahami seberapa dekat prediksi model terhadap kenyataan [illegible] mampu mengungkapkan pola-pola temporal yang mungkin tidak [illegible]n dalam metrik numerik. Dengan memanfaatkan visualisasi ini, [illegible]masing-masing arsitektur model terhadap dinamika fluktuasi [illegible]liamati secara eksplisit, termasuk kecenderungan model untuk [illegible]overfitting pada rentang waktu tertentu. Oleh karena itu, bagian [illegible]s mendalam atas grafik perbandingan antara harga aktual dan

hasil prediksi dari masing-masing model, sebagai bentuk validasi visual terhadap performa yang sebelumnya telah dikaji melalui metrik evaluasi kuantitatif.

Analisis visual terhadap performa model dalam memprediksi harga saham Google, sebagaimana disajikan pada Gambar 2.4, memperlihatkan bahwa hampir seluruh model mampu merekonstruksi tren harga aktual dengan tingkat presisi yang tinggi. Garis prediksi dari model-model deep learning seperti LSTM-P, 2LSTM-P, dan LSTM-PC-LSTM secara konsisten mengikuti pola fluktuasi harga riil, baik pada periode tren naik, penurunan, maupun konsolidasi harga. Kedekatan kurva prediksi terhadap garis harga aktual ini mencerminkan kemampuan model dalam menginternalisasi pola temporal dan dinamika nonlinier pasar saham. Meskipun sebagian model konvensional seperti MLP dan SVR juga menunjukkan kecenderungan mengikuti arah tren, namun akurasinya terlihat menurun pada titik-titik ekstrem, seperti puncak harga dan titik balik, yang mengindikasikan keterbatasan model non-sekuensial dalam menangkap ketergantungan temporal jangka panjang. Sebaliknya, model paralel dengan input multivariat menawarkan keunggulan adaptif yang lebih kuat, sesuai dengan temuan evaluasi kuantitatif sebelumnya. Oleh karena itu, Gambar 2.4 tidak hanya memberikan validasi visual atas performa numerik yang telah dihitung, tetapi juga menegaskan pentingnya pemilihan arsitektur dan strategi penyusunan input dalam mengoptimalkan akurasi prediksi saham berbasis deep learning.



Gambar 2.4 Perbandingan antara Harga Aktual Saham Google dengan Prediksi dari Model MLP, SVR, Vanilla RNN, dan Stacked LSTM Berdasarkan Konfigurasi Input Fitur

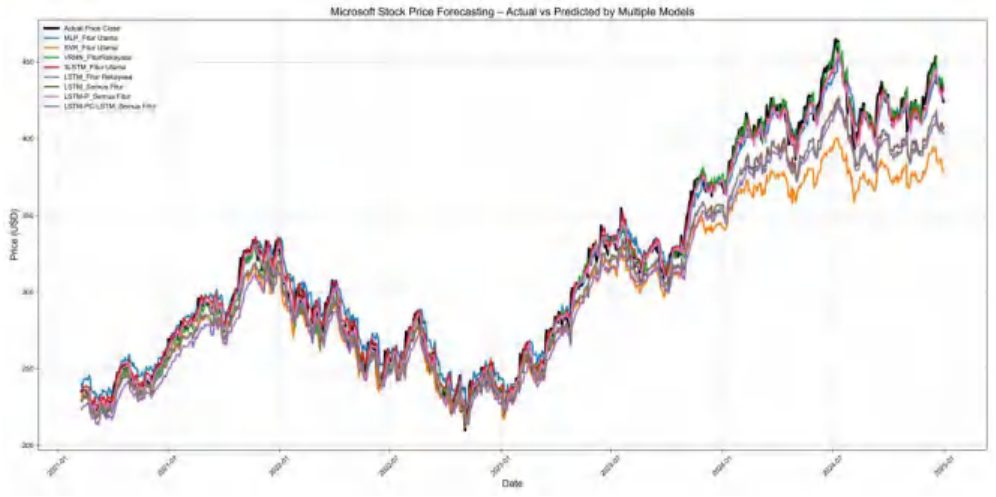


Gambar 2.5 Perbandingan antara Harga Aktual Saham Apple dengan Prediksi dari Model MLP, SVR, Vanilla RNN, dan Stacked LSTM Berdasarkan Konfigurasi Input Fitur

Gambar 2.5, menunjukkan performa model dalam memprediksi harga saham Apple dengan disparitas yang lebih nyata antar arsitektur, terutama saat menghadapi fase-fase harga yang sangat dinamis. Meskipun sebagian besar model mampu mengikuti arah tren jangka panjang, tidak semua berhasil mereplikasi amplitudo fluktuasi harga dengan presisi tinggi. Model-model seperti LSTM-P dan 2LSTM-P memperlihatkan kesesuaian visual yang baik terhadap pola aktual, termasuk pada titik-titik perubahan arah yang tajam. Sebaliknya, model konvensional seperti MLP dan SVR dengan fitur rekayasa tampak tertinggal dalam menangkap momentum pasar, dengan deviasi yang cukup signifikan pada fase lonjakan harga. Hal ini menunjukkan bahwa pendekatan berbasis pemrosesan paralel lebih adaptif dalam menyerap kompleksitas temporal dan heterogenitas fitur input. Di sisi lain, meskipun model 3LSTM berbasis fitur utama sempat mendekati pola aktual pada fase stabil, performanya menurun ketika menghadapi fluktuasi tajam, kemungkinan akibat akumulasi noise internal. Secara keseluruhan, visualisasi ini memperkuat hasil evaluasi kuantitatif sebelumnya bahwa kemampuan generalisasi model tidak hanya ditentukan oleh arsitektur yang dalam, tetapi juga oleh bagaimana fitur diklasifikasikan dan dialirkan dalam struktur jaringan secara modular.

Gambar 2.6, menunjukkah hasil visualisasi prediksi harga saham Microsoft dengan variasi performa antar model yang semakin mencolok, terutama saat memasuki periode tren naik yang tajam. Kurva prediksi dari model-model deep
.STM-P dan 2LSTM-P, tampak lebih koheren mengikuti lintasan
pada titik-titik akselerasi maupun koreksi harga. Kesesuaian ini
mpuan adaptif model paralel dalam menangani struktur data
pleks. Sebaliknya, model seperti SVR dan MLP dengan fitur
ıtkan deviasi yang konsisten terhadap data aktual, khususnya
ın lembah harga, yang mengindikasikan keterbatasan model
enavigasi dinamika temporal yang nonlinier. Sementara model

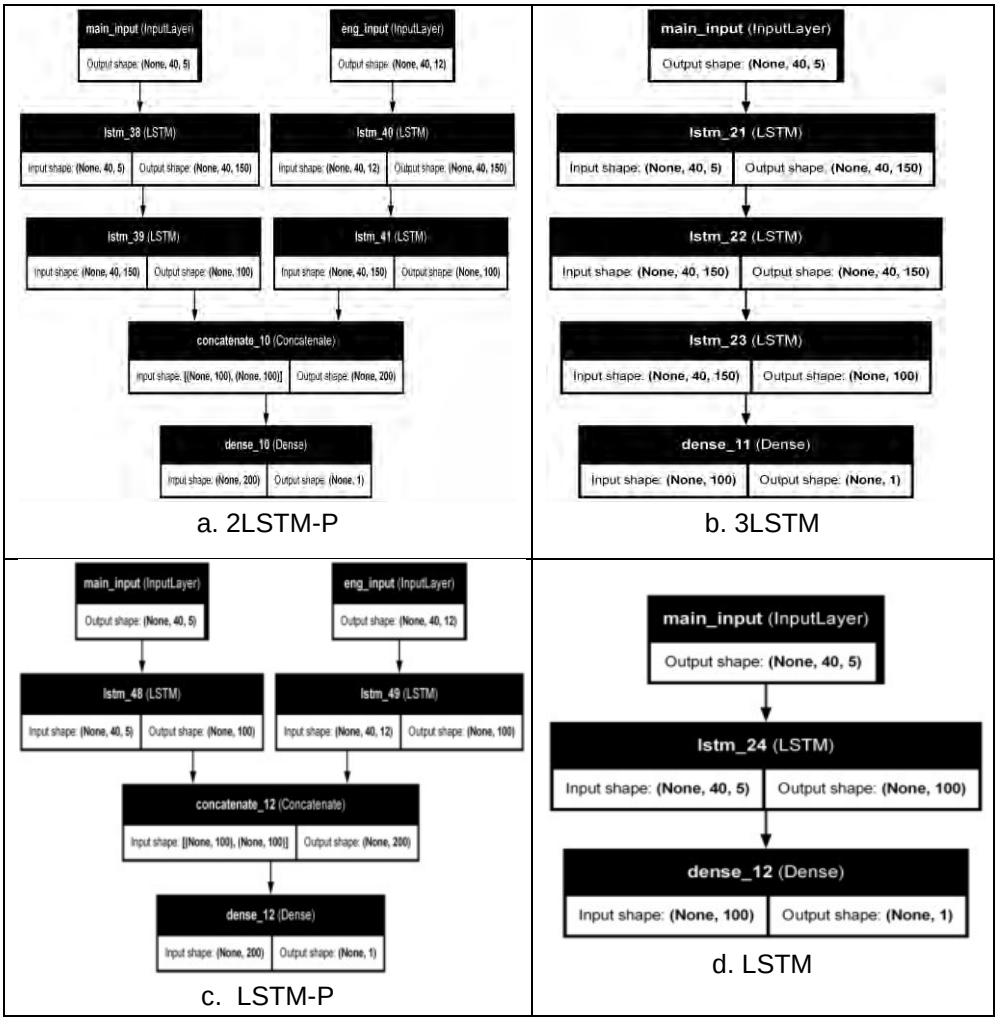
3LSTM menunjukkan performa relatif baik pada rentang waktu menengah, grafiknya mulai menyimpang pada periode fluktuasi tinggi, kemungkinan besar akibat akumulasi error dan overfitting terhadap fitur tambahan. Secara keseluruhan, pola visual ini memberikan gambaran tambahan yang melengkapi hasil kuantitatif: bahwa presisi arsitektur tidak hanya ditentukan oleh kedalaman jaringan, tetapi juga oleh strategi pengelolaan fitur serta kemampuan model menginterpretasi keterkaitan temporal dalam konteks pasar yang cenderung fluktuatif seperti saham Microsoft.



Gambar 2.6 Perbandingan antara Harga Aktual Saham Microsoft dengan Prediksi dari Model MLP, SVR, Vanilla RNN, dan Stacked LSTM Berdasarkan Konfigurasi Input Fitur

Secara umum, analisis prediksi harga saham dari ketiga emiten menunjukkan bahwa model berbasis LSTM dengan arsitektur paralel memberikan performa paling konsisten dan akurat. Pada saham Google, model 2LSTM-P menjadi yang terbaik karena menghasilkan error yang paling rendah dan prediksi visual yang paling mendekati harga aktual, mencerminkan keunggulan adaptif terhadap pola fluktuatif namun stabil, struktur plot model dari 2LSTM-P ditunjukkan pada Gambar 2.7.a. Meskipun disisi model 3LSTM berbasis input tunggal menonjol di antara varian stacked-LSTM karena tetap stabil dan akurat saat konfigurasi input merupakan fitur utama, dimana ilustrasi model seperti pada Gambar 2.7.b. Untuk saham Apple, LSTM-P menunjukkan hasil paling optimal, baik secara numerik maupun visual, terutama dalam menangani dinamika harga yang lebih tajam, menandakan efektivitas pemrosesan fitur multivariat secara terpisah, plot model divisualisasikan pada Gambar 2.7.c. Selain itu, model LSTM original yang
a terbaik dengan input fitur utama pada saham Apple, sepeti
odel pada Gambar 2.7.d.
u, saham Microsoft menunjukkan bahwa LSTM-P tetap unggul
dengan error terkecil dan bias prediksi yang sangat rendah
ıya dengan plot model seperti yang diilustrasikan pada Gambar
emikian, model 3LSTM berbasis input tunggal sangat

diperhitungkan di antara varian stacked-LSTM karena tetap stabil dan akurat saat kompleksitas input dikendalikan. Temuan ini mempertegas bahwa pemilihan model terbaik harus mempertimbangkan interaksi antara kedalaman arsitektur, strategi input, dan karakteristik temporal dari masing-masing saham, bukan semata-mata berdasarkan kompleksitas jaringan.



Gambar 2.7 Plot Model optimal: a dan c Refresentasi dari Input Paralel, b dan d sebagai input Tunggal

ediksi harga saham dari berbagai model memperkuat temuan sitektur berbasis LSTM, khususnya dengan input paralel, ıg paling mendekati harga aktual. Perbedaan pola antar model akurasi prediksi sangat dipengaruhi oleh strategi pengelolaan tektur. Dengan demikian, visualisasi ini menjadi bukti penting del yang tepat harus mempertimbangkan kesesuaian struktural data saham yang dianalisis.

d. Perbandingan hasil penelitian dengan penelitian terkait

Tabel 2.4 menyajikan perbandingan kinerja model prediksi harga saham dalam penelitian ini dengan beberapa studi terdahulu yang relevan. Hasil penelitian menunjukkan bahwa model LSTM-P dan 2LSTM-P yang digunakan dalam prediksi saham Microsoft, Google, dan Apple secara konsisten mencatatkan nilai R^2 di atas 98% dengan RMSE dan MAE yang relatif rendah, menunjukkan akurasi dan stabilitas prediksi yang tinggi. Sebagai contoh, pada saham Microsoft, LSTM-P menghasilkan R² sebesar 9917%, yang secara signifikan lebih tinggi dibandingkan performa model RNN dan HyRNN dari (John & Latha, 2023a), yang hanya mencatatkan R^2 sebesar 86,7% dan 98,4% meskipun telah mengintegrasikan berita sebagai variabel tambahan. Perbandingan ini menegaskan bahwa efektivitas prediksi tidak hanya bergantung pada banyaknya fitur, tetapi lebih ditentukan oleh arsitektur model dan strategi pengelolaan input.

Pada saham Google, model 2LSTM-P dan 3LSTM berbasis input tunggal juga menunjukkan performa yang kompetitif, dengan nilai RMSE masing-masing sebesar 3.4765 dan 2.7401, serta R^2 sebesar 98.12% dan 98.83%. Nilai ini setara bahkan sedikit lebih tinggi dari performa VMD–TMFG–LSTM dalam studi Zhang et al. (2025) yang mencatatkan R^2 sebesar 97.80%, meskipun model tersebut menggunakan kombinasi teknik dekomposisi dan filtering canggih. Fakta ini memperkuat bahwa arsitektur stacked-LSTM yang tepat, bahkan tanpa teknik tuning kompleks, dapat memberikan hasil yang unggul apabila struktur input dikelola secara efektif.

Pendekatan LSTM-P pada prediksi Saham Apple juga menunjukkan hasil yang sangat baik (R^2 = 98.65%), sebanding dengan model 1-Layer LSTM dari Konur et al., (2024) yang diterapkan pada saham-saham Turki dan mencatatkan R^2 antara 97.29% hingga 98.86%. Namun demikian, perlu dicatat bahwa skala harga saham dan volatilitas pasar tempat penelitian dilakukan sangat mempengaruhi nilai RMSE absolut, sehingga perbandingan harus lebih fokus pada rasio R^2 dan efisiensi struktural model.

Sementara itu, studi Dwiandiyanta et al. (2025) pada saham domestik Indonesia (BBCA, BBRI, BMRI, TLKM, ASII, dan BBNI) menunjukkan R^2 berkisar antara 88.54% hingga 96.78%. Meski nilai RMSE-nya rendah (misalnya BBCA = 0.0289), hal tersebut dipengaruhi oleh skala harga yang kecil, sehingga tidak dapat dibandingkan langsung secara absolut. Namun dari sisi proporsionalitas, model LSTM dan GRU tersebut belum mencapai tingkat akurasi prediktif seperti pada pendekatan LSTM-P dalam penelitian ini.

enunjukkah bahwa model dalam penelitian ini tidak hanya
li atau menyamai performa studi terdahulu, tetapi juga
si signifikan melalui pengembangan baseline prediktif berbasis
tuning. Keunggulan ini diperoleh berkat struktur input paralel
hkan dan mengelola representasi fitur utama dan fitur rekayasa
ngga menghasilkan generalisasi prediksi yang kuat pada
saham.

Tabel 2.4 Perbandingan Kinerja Model Prediksi Harga Saham dengan Penelitian Sebelumnya Berdasarkan Evaluasi Kuantitatif

Peneliti (Tahun)	Model	Fitur	Saham	RMSE	MAE (%)	R^2 (%)
Penelitian Ini (2025)	LSTM-P	Utama + Rekayasa	Microsoft	6.3058	1.57	99.17
	2LSTM-P	Utama + Rekayasa	Google	3.4765	2.08	98.12
	LSTM-P	Utama + Rekayasa	Apple	3.6191	1.72	98.65
	3LSTM	Utama	Google	2.7401	2.01	98.83
John & Latha. (2023)	RNN	Utama	Nasdaq	52.933	52.933	86.7
	RNN-LSTM	Utama		23.07	23.07	97.9
	RNN-GRU	Utama		20.752	20.752	98.1
	HyRNN	Utama		19.512	19.512	98.4
	RNN	Utama + Berita		49.824	46.17	87.1
	RNN-LSTM	Utama + Berita		21.87	15.489	98.1
	RNN-GRU	Utama + Berita		19.152	14.321	98.3
	HyRNN	Utama + Berita		18.142	13.102	98.7
Dwiandiyanta et al. (2025)	RNN	Utama + Rekayasa	BBCA	0.0289	-	93.84
	RNN	Utama + Rekayasa	BBRI	0.0275	-	88.54
	RNN	Utama + Rekayasa	BMRI	0.0345	-	96.78
	RNN	Utama + Rekayasa	TLKM	0.0342	-	91.26
	LSTM	Utama + Rekayasa	ASII	0.0216	-	94.77
	GRU	Utama + Rekayasa	BBNI	0.0242	-	94.67
Zhang et al. (2025)	TMFG–LSTM	Utama + Rekayasa	Shanghai International Airport Co., Ltd. (sh600009)	2.3046	1.6327	97.09
	VMD–LSTM	Utama + Rekayasa		5.6986	4.2675	82.2
	VMD–TMFG–LSTM	Utama + Rekayasa		2.0029	1.4677	97.8
Konur et al. (2024)	1-Layer LSTM	Utama + Rekayasa	FROTO.IS	0.0266	0.0205	98.65
	1-Layer LSTM	Utama + Rekayasa	ISCTR.IS	0.0362	0.0279	97.29
	1-Layer LSTM	Utama + Rekayasa	KCHOL.IS	0.0265	0.0209	98.68
	1-Layer LSTM	Utama + Rekayasa	THYAO.IS	0.0311	0.024	98.18
	1-Layer LSTM	Utama + Rekayasa	TUPRS.IS	0.0231	0.0165	98.86

e. Temuan Penelitian

Penelitian ini menghasilkan beberapa temuan penting yang menunjukkan efektivitas pendekatan deep learning berbasis stacked LSTM dalam prediksi harga saham multivariat. Pertama, konfigurasi input paralel yang memisahkan jalur fitur utama dan fitur rekayasa terbukti meningkatkan stabilitas prediksi serta menurunkan nilai error secara signifikan dibandingkan konfigurasi input tunggal atau gabungan tanpa pengelolaan alur representasi. Model LSTM-P dan 2LSTM-P secara konsisten

ma unggul pada ketiga saham yang dianalisis, dengan hingga 10–12% dan penurunan bias prediksi lebih dari 50% baseline seperti MLP, SVR, dan Vanilla RNN.

tur arsitektur model memengaruhi sensitivitas terhadap input. ut tunggal berbasis fitur utama, model 3LSTM menghasilkan k saham Google, menunjukkan bahwa peningkatan kedalaman lam menangkap pola temporal jangka panjang selama

kompleksitas input dikendalikan. Sebaliknya, model dengan arsitektur dalam seperti 3LSTM-PC-LSTM cenderung mengalami degradasi performa ketika diberi input gabungan tanpa struktur, yang menunjukkan indikasi overfitting akibat propagasi fitur yang tidak relevan.

Ketiga, hasil evaluasi kuantitatif diperkuat oleh analisis visualisasi, yang menunjukkan bahwa model terbaik tidak hanya mendekati harga aktual secara statistik, tetapi juga berhasil mengikuti pola fluktuasi pasar secara temporal. Hasil ini didukung oleh korelasi tinggi antara data aktual dan hasil prediksi ($r > 0.98$), serta nilai PBIAS yang mendekati nol pada model LSTM-P, yang mengindikasikan kestabilan bias prediksi dalam konteks pasar saham yang dinamis.

Keempat, perbandingan dengan studi terdahulu menunjukkan bahwa model dalam penelitian ini mampu menyamai bahkan melampaui performa model hybrid yang lebih kompleks, seperti VMD–TMFG–LSTM atau HyRNN dengan integrasi berita, dengan konfigurasi yang lebih sistematis dan efisien. Oleh karena itu, pendekatan yang digunakan dalam penelitian ini dapat dijadikan baseline kuat untuk pengembangan metode prediksi lanjutan yang mencakup optimasi hyperparameter, attention mechanism, dan integrasi sistem real-time berbasis AI dalam konteks keuangan.

f. Keterbatasan dan Penelitian Lanjutan

Penelitian ini memiliki beberapa keterbatasan yang perlu dicermati untuk pengembangan lebih lanjut. Pertama, meskipun eksperimen telah mencakup berbagai konfigurasi arsitektur dan strategi input, pemodelan masih terbatas pada data historis berbasis teknikal, tanpa melibatkan sentimen pasar, faktor makroekonomi, atau peristiwa eksternal yang dapat memengaruhi harga saham secara signifikan. Kedua, meskipun pendekatan paralel telah menunjukkan keunggulan, kompleksitas model meningkat secara substansial seiring bertambahnya jalur input, yang berpotensi menimbulkan tantangan komputasi pada implementasi real-time atau skenario berbasis edge computing. Ketiga, pengujian dilakukan pada data tiga saham besar (Google, Apple, Microsoft) yang memiliki likuiditas tinggi dan pola yang relatif stabil, sehingga generalisasi terhadap saham dengan volatilitas ekstrem atau kapitalisasi kecil masih perlu divalidasi.

Sebagai arah penelitian lanjutan, disarankan pengembangan model prediksi yang menggabungkan fitur teknikal dan eksternal (seperti berita, volume sentimen, atau indikator ekonomi) dalam arsitektur multimodal yang lebih kompleks. Penggunaan algoritma optimasi adaptif, seperti Bayesian Optimization atau Hybrid Metaheuristik, juga berpotensi meningkatkan efisiensi tuning hyperparameter secara
u, eksplorasi model berbasis attention mechanism atau
menangkap ketergantungan jangka panjang dapat memperluas
uasi performa juga dapat diperluas dengan menguji model pada
erangka time horizon yang berbeda untuk menguji robustitas
ariasi pasar. Dengan mempertimbangkan arah ini, penelitian
emberikan kontribusi yang lebih aplikatif dalam pengembangan

sistem prediksi keuangan berbasis kecerdasan buatan yang akurat, adaptif, dan responsif terhadap dinamika pasar.

2.5 Kesimpulan

Berdasarkan uraian yang telah dijelaskan dari hasil penelitian dan pembahasan, penelitian ini menunjukkan bahwa arsitektur deep learning, khususnya model LSTM dengan konfigurasi paralel dan strategi input yang terstruktur, memiliki keunggulan yang konsisten dalam memprediksi harga saham multivariabel. Hasil evaluasi kuantitatif dan visual terhadap saham Google, Apple, dan Microsoft mengonfirmasi bahwa model-model seperti LSTM-P dan 2LSTM-P mampu memberikan prediksi yang paling akurat dan stabil, terutama dalam mengelola data dengan kombinasi fitur utama dan fitur rekayasa. Namun demikian, pada konfigurasi input tunggal, model stacked-LSTM seperti 3LSTM juga terbukti efektif ketika diterapkan pada saham dengan pola harga yang relatif stabil. Visualisasi prediksi mendukung temuan ini dengan menunjukkan kesesuaian yang tinggi antara kurva prediksi dan harga aktual pada model-model yang memiliki desain arsitektur dan strategi input yang selaras. Dengan demikian, penelitian ini menegaskan pentingnya pemilihan arsitektur dan desain input sebagai faktor kunci dalam membangun model prediksi saham berbasis deep learning yang presisi, sekaligus menyediakan baseline objektif untuk penelitian lanjutan yang akan mengintegrasikan optimasi lanjutan dan teknik lainnya.

2.6 Daftar Pustaka

Aseeri, A. O. (2023). Effective short-term forecasts of Saudi stock price trends using technical indicators and large-scale multivariate time series. *PeerJ Computer Science*, *9*, e1205. https://doi.org/10.7717/PEERJ-CS.1205/SUPP-3

Bhuiyan, M. S. M., Rafi, M. AL, Rodrigues, G. N., Mir, M. N. H., Ishraq, A., Mridha, M. F., & Shin, J. (2025). Deep learning for algorithmic trading: A systematic review of predictive models and optimization strategies. *Array*, *26*, 100390. https://doi.org/10.1016/J.ARRAY.2025.100390

Das, N., Sadhukhan, B., Ghosh, R., & Chakrabarti, S. (2024). Developing Hybrid Deep Learning Models for Stock Price Prediction Using Enhanced Twitter Sentiment Score and Technical Indicators. *Computational Economics*. https://doi.org/10.1007/S10614-024-10566-9

Dwiandiyanta, B. Y., Hartanto, R., & Ferdiana, R. (2025). Harnessing Deep Learning and Technical Indicators for Enhanced Stock Predictions of Blue-Chip Stocks on the Indonesia Stock Exchange (IDX). *Engineering, Technology & Applied Science Research*, *15*(1), 20348–20357. https://doi.org/10.48084/ETASR.9850

Fjellstrom, C. (2022). Long Short-Term Memory Neural Network for Financial Time Series. *Proceedings - 2022 IEEE International Conference on Big Data, Big Data 2022*, 3496–3504. https://doi.org/10.1109/BigData55660.2022.10020784

Gupta, P., Kumar Gupta, S., & Singh Jadon, R. (2023). Stock Market Prediction using RNN-based Models with Random and Tuned Hyperparameters. *International Journal of Computer Applications*, *185*(21), 12–17. https://doi.org/10.5120/ijca2023922935

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, *9*(8), 1735–1780. https://doi.org/10.1162/NECO.1997.9.8.1735

Hong, J., Han, P., Rasool, A., Chen, H., Hong, Z., Tan, Z., Lin, F., Wei, S. X., & Jiang, Q. (2023). A Correlational Strategy for the Prediction of High-Dimensional Stock Data by Neural Networks and Technical Indicators. *Communications in Computer and Information Science*, *1796 CCIS*. https://doi.org/10.1007/978-981-99-3300-6_29

Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice*. OTexts

(2023). Stock market prediction based on a deep hybrid RNN sentiment analysis. *Automatika*, *64*(4), 981–995. D.1080/00051144.2023.2217602;WGROUP:STRING:PUBLIC

Karim, M. E., Foysal, M., & Das, S. (2023). Stock Price Prediction Using Bi-LSTM and GRU-Based Hybrid Deep Learning Approach. *Lecture Notes in Networks and Systems*, *479*, 701–711. https://doi.org/10.1007/978-981-19-3148-2_60

Konur, M., Göçken, M., & Dosdoğru, A. T. (2024). Stock Price Prediction Using Deep Learning Algorithms Based on Technical Indicators. *Journal of Operations Intelligence*, *2*(1), 300–320. https://doi.org/10.31181/JOPI21202427.

Lawi, A., Mesra, H., & Amir, S. (2022). Implementation of Long Short-Term Memory and Gated Recurrent Units on grouped time-series data to predict stock prices accurately. *Journal of Big Data*, *9*(1). https://doi.org/10.1186/s40537-022-00597-0

Long, W., Lu, Z., & Cui, L. (2019). Deep learning-based feature engineering for stock price movement prediction. *Knowledge-Based Systems*, *164*, 163–173. https://doi.org/10.1016/J.KNOSYS.2018.10.034

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and Machine Learning forecasting methods: Concerns and ways forward. *PLoS ONE*, *13*(3). https://doi.org/10.1371/JOURNAL.PONE.0194889

Moodi, F., Rafsanjani, A. J., Zarifzadeh, S., & Chahooki, M. A. Z. (2024). Fusion of technical indicators and sentiment analysis in a hybrid framework of deep learning models for stock price movement prediction. *IEEE Access*. https://doi.org/10.1109/ACCESS.2024.3519556

Noh, Y., Kim, J.-M., Hong, S., & Kim, S. (2023). Deep Learning Model for Multivariate High-Frequency Time-Series Data: Financial Market Index Prediction. *Mathematics 2023, Vol. 11, Page 3603*, *11*(16), 3603. https://doi.org/10.3390/MATH11163603

Rataj, M., Zhang, X., Wang, J.-Q., Shantal, M., Othman, Z., Abu Bakar, A., & My, A. A. B. (2023). A Novel Approach for Data Feature Weighting Using Correlation Coefficients and Min–Max Normalization. *Symmetry*, *15*(12), 2185. https://doi.org/10.3390/SYM15122185

Saberironaghi, M., Ren, J., & Saberironaghi, A. (2025). Stock Market Prediction Using Machine Learning and Deep Learning Techniques: A Review. *AppliedMath 2025, Vol. 5, Page 76*, *5*(3), 76. https://doi.org/10.3390/APPLIEDMATH5030076

Samantaray, S., Sahoo, A., Yaseen, Z. M., & Al-Suwaiyan, M. S. (2025). River diction based multivariate climatological variables using short-term memory with nature inspired algorithm. *Journal of* 132453. https://doi.org/10.1016/J.JHYDROL.2024.132453

tia, D. V. (2025). Multi-variate LSTM with attention mechanism tock market. *International Journal of Information Management* (2), 100350. https://doi.org/10.1016/J.JJIMEI.2025.100350

Shao, S. (2024). Stacked Block Analysis Based on LSTM for Stock Price Prediction. In *Transactions on Computer Science and Intelligent Systems Research* (Vol. 5).

Singh, U., Tamrakar, S., Saurabh, K., Vyas, R., & Vyas, O. P. (2024). Optimizing Hyperparameters of Deep Learning Models for Stock Price Prediction. *15th International Conference on Computing, Communication, and Networking Technologies (ICCCNT)*. https://doi.org/10.1109/ICCCNT61001.2024.10724813

Tuan, N. T., Nguyen, T. H., & Duong, T. T. H. (2022). Stock Price Prediction in Vietnam Using Stacked LSTM. *Lecture Notes on Data Engineering and Communications Technologies*, *148*, 246–255. https://doi.org/10.1007/978-3-031-15063-0_23

Wen, X., & Li, W. (2023). Time Series Prediction Based on LSTM-Attention-LSTM Model. *IEEE Access*, *11*. https://doi.org/10.1109/ACCESS.2023.3276628

Xu, Y., Chhim, L., Zheng, B., & Nojima, Y. (2020). Stacked Deep Learning Structure with Bidirectional Long-Short Term Memory for Stock Market Prediction. *Communications in Computer and Information Science*, *1265 CCIS*, 447–460. https://doi.org/10.1007/978-981-15-7670-6_37

Zhang, Z., Liu, Q., Hu, Y., & Liu, H. (2025). Multi-feature stock price prediction by LSTM networks based on VMD and TMFG. *Journal of Big Data*, *12*(1). https://doi.org/10.1186/S40537-025-01127-4

Zhou, Q., Zhou, C., & Wang, X. (2022). Stock prediction based on bidirectional gated recurrent unit with convolutional neural network and feature selection. *PLoS ONE*, *17*(2 February). https://doi.org/10.1371/journal.pone.0262501