BAB II
KAJIAN LITERATUR

2.1 Pasar Modal dan Harga Saham

Pasar modal adalah tempat bertemunya pihak yang membutuhkan dana dengan pihak yang ingin menanamkan dananya. Perusahaan yang membutuhkan dana dapat menjual sebagian kepemilikannya dalam bentuk saham, sedangkan pemodal membeli saham tersebut dengan harapan memperoleh keuntungan di kemudian hari. Saham adalah surat berharga yang menjadi bukti kepemilikan atas sebagian kecil sebuah perusahaan [2]. Harga saham tidak ditetapkan oleh satu pihak, melainkan terbentuk dari tawar-menawar antara pembeli dan penjual di bursa. Ketika lebih banyak orang ingin membeli daripada menjual, harga cenderung naik, dan sebaliknya harga cenderung turun ketika penjual lebih banyak. Karena tawar-menawar itu berlangsung terus-menerus selama jam perdagangan, harga saham berubah setiap hari bahkan setiap menit. Pergerakan yang terus berubah inilah yang membuat harga saham menarik sekaligus sulit diperkirakan.

Selain harga penutupan biasa, bursa juga mencatat harga penutupan yang telah disesuaikan atau *adjusted close*. Penyesuaian ini diperlukan karena perusahaan kadang melakukan tindakan tertentu, misalnya membagi satu lembar saham menjadi beberapa lembar atau membagikan keuntungan kepada pemegang saham. Tanpa penyesuaian, tindakan tersebut akan terlihat seolah-olah harga turun drastis padahal nilai kepemilikan pemodal tidak berkurang. Harga yang telah disesuaikan membuat deret harga dari tahun ke tahun dapat dibandingkan secara adil [16]. Karena alasan itu, penelitian ini memakai harga penutupan yang telah disesuaikan sebagai nilai yang diprediksi. Pemilihan ini juga menjaga agar model tidak salah belajar dari penurunan harga yang sebenarnya hanya akibat pemecahan saham. Dengan demikian, data yang dipakai mencerminkan perubahan nilai yang sesungguhnya dirasakan pemodal. Grafik historis harga penutupan yang disesuaikan untuk saham GOOGL sepanjang periode data tersebut dapat dilihat pada Gambar 2.1.

Gambar 2.1 Grafik Historis Harga Penutupan Disesuaikan Saham GOOGL [3]

2.2 Prediksi Harga Saham dan Data Deret Waktu

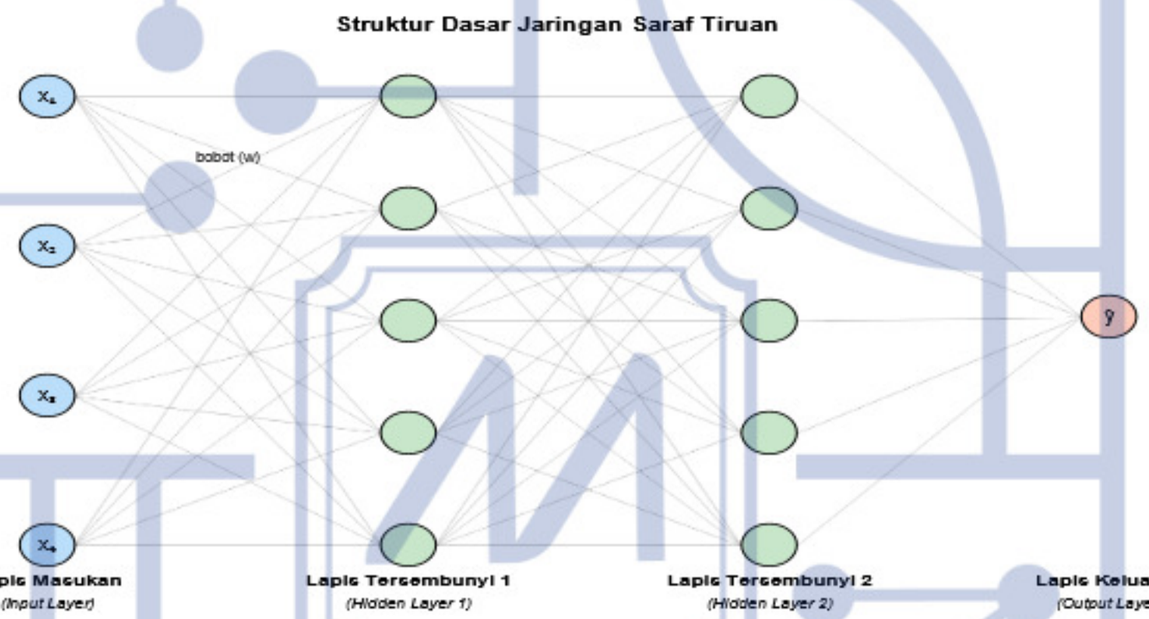
Data deret waktu adalah data yang dicatat berurutan menurut waktu, misalnya harga saham harian dari tahun ke tahun. Ciri utama data jenis ini adalah urutannya tidak boleh diacak, sebab nilai hari ini berkaitan dengan nilai hari-hari sebelumnya. Peramalan deret waktu berarti memperkirakan nilai yang akan datang dengan mempelajari pola pada nilai masa lalu [15]. Pola yang biasa dicari meliputi kecenderungan naik atau turun dalam jangka panjang, pengulangan musiman, serta guncangan mendadak yang sulit ditebak. Pada harga saham, pola musiman jarang terlihat jelas, sedangkan guncangan mendadak justru sering terjadi. Hal itu membuat peramalan harga saham tergolong lebih sulit dibandingkan peramalan data lain seperti penjualan bulanan. Karena itu, hasil peramalan harga saham perlu dinilai dengan hati-hati dan tidak boleh dianggap sebagai kepastian.

Sebuah deret waktu disebut stabil apabila sifat-sifatnya, seperti nilai rata-rata dan besar naik-turunnya, tidak berubah banyak sepanjang waktu. Harga saham umumnya tidak stabil karena nilainya dapat tumbuh berkali-kali lipat dalam rentang bertahun-tahun. Model peramalan yang dilatih langsung pada harga yang terus menanjak berisiko hanya menghafal kecenderungan naik, bukan mempelajari pola perubahannya [4]. Untuk mengatasi hal tersebut, deret harga biasanya diubah lebih dahulu menjadi bentuk perubahan harian, sehingga sifatnya menjadi lebih stabil. Cara pengubahan yang dipakai pada penelitian ini dijelaskan pada Subbab 2.7. Dengan deret yang lebih stabil, model dapat memusatkan perhatian pada pola perubahan harga, bukan pada tingkat harganya.

2.3 Pembelajaran Mesin dan Pembelajaran Mendalam

Pembelajaran mesin adalah cara membuat komputer mengenali pola dari data tanpa aturan yang ditulis satu per satu oleh manusia. Komputer diberi banyak contoh, lalu

menyesuaikan perhitungan di dalamnya sampai keluarannya mendekati jawaban yang benar [13]. Proses penyesuaian itu disebut pelatihan, dan data yang dipakai disebut data latih. Setelah dilatih, kemampuan model diuji memakai data lain yang belum pernah dilihatnya, agar ketahuan apakah model benar-benar belajar atau sekadar menghafal. Pembelajaran mendalam adalah cabang pembelajaran mesin yang memakai jaringan saraf tiruan berlapis banyak. Susunan berlapis membuat model mampu mengenali pola yang bertingkat dan tidak lurus [6]. Karena itu pembelajaran mendalam banyak dipakai pada masalah yang polanya rumit, termasuk peramalan harga saham. Struktur dasar jaringan saraf tiruan dengan lapis masukan, lapis tersembunyi, dan lapis keluaran digambarkan secara umum pada Gambar 2.2.



Gambar 2.2 Struktur Dasar Jaringan Saraf Tiruan [6]

Jaringan saraf tiruan tersusun atas unit-unit hitung kecil yang disebut neuron dan dikelompokkan dalam beberapa lapis. Lapis masukan menerima data, lapis tersembunyi mengolah data tersebut, dan lapis keluaran menghasilkan jawaban akhir. Setiap sambungan antarneuron memiliki bobot, yaitu angka yang menentukan seberapa besar pengaruh satu neuron terhadap neuron berikutnya. Nilai bobot inilah yang diubah-ubah selama pelatihan sampai kesalahan model menjadi sekecil mungkin. Agar jaringan dapat mempelajari pola yang tidak lurus, keluaran tiap neuron dilewatkan pada fungsi pengaktif. Fungsi pengaktif membuat jaringan tidak sekadar menjumlahkan angka secara lurus, melainkan mampu membentuk hubungan yang melengkung dan bertingkat [6]. Tanpa fungsi pengaktif, penambahan lapis tidak akan menambah kemampuan jaringan.

Fungsi pengaktif ReLU dinyatakan pada Persamaan (2.1a).

$$f(x) = \max(0, x) \quad \text{.....(2.1a)}$$

Dimana:

x = nilai masukan neuron, dan

f(x) = keluaran sesudah pengaktifan, bernilai nol untuk seluruh masukan negatif.

Fungsi pengaktif sigmoid dinyatakan pada Persamaan (2.2a).

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad \text{.....(2.2a)}$$

Dimana:

σ = fungsi sigmoid yang selalu menghasilkan nilai antara nol dan satu, dan

e = bilangan pokok logaritma alami.

Fungsi pengaktif tangen hiperbolik dinyatakan pada Persamaan (2.4a).

$$\tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}} \quad \text{.....(2.4a)}$$

Dimana:

tanh = fungsi tangen hiperbolik yang menghasilkan nilai antara negatif satu dan satu.

Sigmoid dipakai pada gerbang pelupa dan gerbang masukan Persamaan (2.2) dan (2.3) serta pada kedua gerbang GRU Persamaan (2.7a) dan (2.7b), sedangkan tangen hiperbolik dipakai pada Persamaan (2.4), (2.6), dan (2.7c).

2.4 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) adalah arsitektur jaringan saraf tiruan yang pada awalnya populer dalam pengolahan citra, namun kemudian terbukti efektif pula untuk data deret waktu. Secara umum, CNN tersusun atas tiga kelompok lapisan utama, yaitu lapisan konvolusi yang bertugas mengekstraksi pola lokal dari data masukan, lapisan penggabung (pooling) yang meringkas informasi untuk mengurangi dimensi, dan lapisan padat (dense) yang menggabungkan seluruh representasi fitur untuk menghasilkan keluaran akhir. Dalam penelitian ini, CNN digunakan untuk mengenali pola jangka pendek pada deret log-return harga saham GOOGL yang disusun dalam jendela geser. Kemampuan CNN mengekstraksi pola setempat menjadikannya kandidat yang relevan untuk dibandingkan dengan arsitektur berbasis memori seperti LSTM dan GRU. Gambaran umum arsitektur CNN secara konseptual dapat dilihat pada Gambar 2.4. Subbab berikut menguraikan mekanisme konvolusi satu dimensi serta komponen-komponen pendukungnya, tanpa menyertakan angka parameter yang bersifat implementasi spesifik karena angka tersebut dibahas pada Bab IV.

2.4.1 Konsep dan Cara Kerja CNN

Convolutional Neural Network adalah jenis jaringan saraf yang bekerja dengan menggeser sebuah jendela kecil di sepanjang data untuk mencari pola setempat [17]. Jendela kecil tersebut disebut penapis, dan isinya berupa sekumpulan bobot yang dipelajari selama

pelatihan. Penapis digeser dari awal sampai akhir data, dan pada tiap posisi ia menghitung satu angka yang menyatakan seberapa kuat pola yang dicarinya muncul di bagian itu. Karena penapis yang sama dipakai ulang di seluruh bagian data, jumlah bobot yang harus dipelajari menjadi jauh lebih sedikit dibandingkan jaringan biasa. Sifat pemakaian ulang ini juga membuat CNN dapat mengenali pola yang sama meskipun letaknya bergeser. Pada data harga saham, pola setempat yang dimaksud dapat berupa kenaikan tajam dalam beberapa hari atau perubahan arah yang mendadak. Kemampuan menangkap pola jangka pendek inilah yang membuat CNN dipertimbangkan untuk peramalan harga. Secara matematis, operasi konvolusi satu dimensi pada satu posisi penapis dinyatakan pada Persamaan (2.1).

$$y[t] = \sum_k (x[t+k] \cdot w[k]) + b \quad \text{.....(2.1)}$$

Dimana:

y = keluaran hasil konvolusi,

x = nilai masukan,

w = bobot penapis,

b = bias, dan

k = panjang penapis.

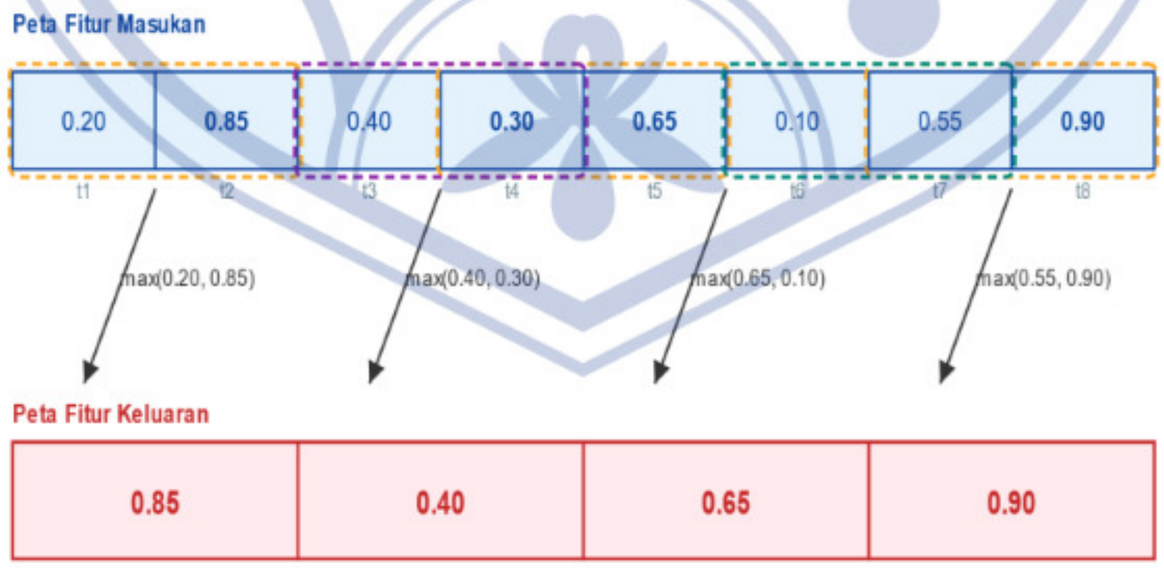
Persamaan ini pada dasarnya mengatakan bahwa keluaran pada setiap posisi dihitung dengan menjumlahkan hasil perkalian antara nilai data masukan dan bobot penapis, lalu menambahkan suku bias. Semakin besar nilai keluaran yang dihasilkan, semakin kuat pola yang dicari penapis tersebut muncul di posisi itu.

Perhitungan pada satu posisi penapis dilakukan dengan mengalikan tiap nilai masukan dengan bobot yang bersesuaian, lalu menjumlahkan seluruh hasil perkalian tersebut. Hasil penjumlahan ditambah sebuah bilangan penggeser yang disebut bias, kemudian dilewatkan pada fungsi pengaktif. Setelah penapis selesai digeser ke seluruh bagian data, terbentuklah barisan angka baru yang disebut peta ciri. Peta ciri menyatakan di bagian mana saja pola yang dicari penapis itu muncul dengan kuat. Sebuah lapis konvolusi umumnya memakai banyak penapis sekaligus, sehingga menghasilkan banyak peta ciri yang berbeda-beda. Dengan cara itu satu lapis dapat mencari beberapa macam pola pada waktu yang bersamaan. Peta-peta ciri tersebut kemudian diteruskan ke lapis berikutnya untuk diolah lebih lanjut.

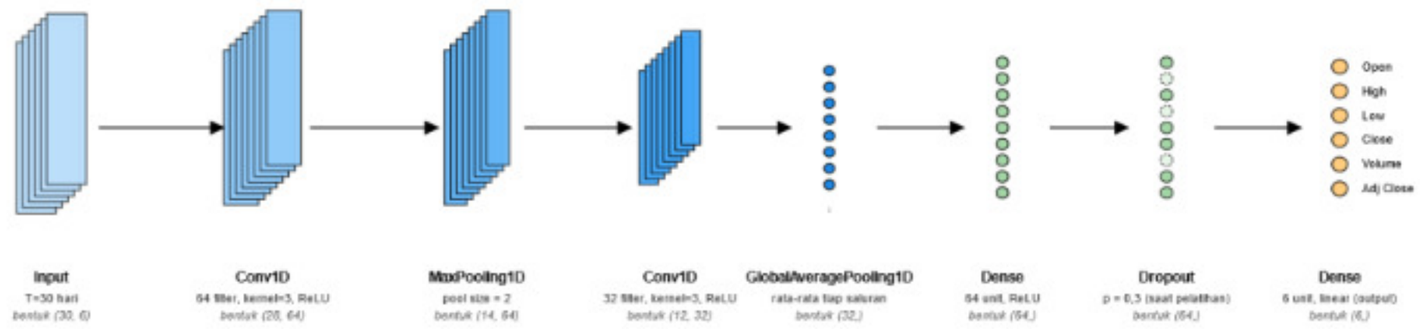
2.4.2 Komponen Arsitektur CNN

Selain lapis konvolusi, CNN memakai beberapa komponen pendukung yang saling melengkapi. Lapis penggabung bertugas meringkas peta ciri dengan mengambil nilai terbesar dari setiap kelompok kecil nilai yang berdekatan. Peringkasan ini memperkecil ukuran data sehingga perhitungan menjadi lebih ringan, sekaligus mempertahankan pola yang paling menonjol. Setelah beberapa lapis konvolusi dan penggabung, hasilnya diteruskan ke lapis padat yang menggabungkan seluruh informasi untuk menghasilkan angka ramalan. Agar model tidak sekadar menghafal data latih, digunakan teknik penjatuhan acak yang mematikan sebagian neuron secara acak selama pelatihan [18]. Teknik tersebut memaksa jaringan tidak bergantung pada segelintir neuron saja.

Susunan lengkap komponen-komponen tersebut membentuk arsitektur CNN yang siap dipakai untuk peramalan. Urutan penyusunannya tidak boleh sembarangan karena setiap lapis bekerja atas keluaran lapis yang berada sebelumnya. Lapis konvolusi diletakkan paling depan agar pola setempat pada deret harga ditemukan lebih dahulu sebelum data diringkas. Lapis penggabung menyusul untuk memperkecil ukuran peta ciri sekaligus mempertahankan pola yang paling menonjol pada setiap bagian. Banyaknya pengulangan pasangan kedua lapis tersebut menentukan seberapa dalam jaringan yang terbentuk dan seberapa luas pola yang dapat ditangkap. Lapis padat ditempatkan di bagian akhir karena bertugas menggabungkan seluruh ciri yang tersisa menjadi satu angka ramalan. Mekanisme lapis penggabung dengan pengambilan nilai terbesar pada setiap kelompok kecil nilai berdekatan dapat dilihat pada Gambar 2.3. Susunan arsitektur CNN yang digunakan untuk peramalan harga saham secara konseptual ditunjukkan pada Gambar 2.4.



Gambar 2.3 Mekanisme Lapis Penggabung dengan Pengambilan Nilai Terbesar [17]



Gambar 2.4 Susunan Arsitektur CNN untuk Peramalan Harga Saham [17]

Banyaknya bobot yang harus dipelajari sebuah lapis dapat dihitung dari ukuran penapis, jumlah masukan, dan jumlah penapis yang dipakai. Pada lapis konvolusi, jumlah bobot diperoleh dengan mengalikan panjang penapis, jumlah saluran masukan, dan jumlah penapis, lalu ditambah satu bias untuk setiap penapis. Pada lapis padat, jumlah bobot diperoleh dengan mengalikan jumlah masukan dengan jumlah keluaran, lalu ditambah satu bias untuk setiap keluaran. Perhitungan tersebut penting karena jumlah bobot menentukan seberapa besar kemampuan model menyimpan pola. Model dengan bobot terlalu banyak dibandingkan jumlah data latih cenderung menghafal, sedangkan model dengan bobot terlalu sedikit sulit menangkap pola. Konsep perhitungan inilah yang dipakai pada Bab IV untuk memeriksa kewajaran ukuran model. Angka hasil perhitungannya sendiri disajikan pada bab tersebut.

2.5 Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) adalah varian jaringan saraf berulang yang dirancang khusus untuk mengatasi keterbatasan jaringan berulang sederhana dalam mengingat informasi jangka panjang. Secara umum, LSTM tersusun atas unit-unit sel memori yang dilengkapi tiga jenis gerbang pengatur, yaitu gerbang pelupa, gerbang masukan, dan gerbang keluaran, yang bersama-sama menentukan aliran informasi di dalam jaringan. Susunan tersebut memungkinkan LSTM mempertahankan informasi yang relevan dalam waktu yang jauh lebih lama dibandingkan jaringan berulang standar. Kemampuan memori jangka panjang ini menjadikan LSTM sangat cocok untuk data deret waktu yang memiliki ketergantungan antarwaktu yang jauh, termasuk pergerakan harga saham. Ilustrasi arsitektur sel LSTM beserta mekanisme gerbangnya secara umum dapat dilihat pada Gambar 2.5. Subbab-subbab berikut menguraikan secara rinci struktur sel memori dan cara kerja setiap gerbang yang membentuk mekanisme inti LSTM.

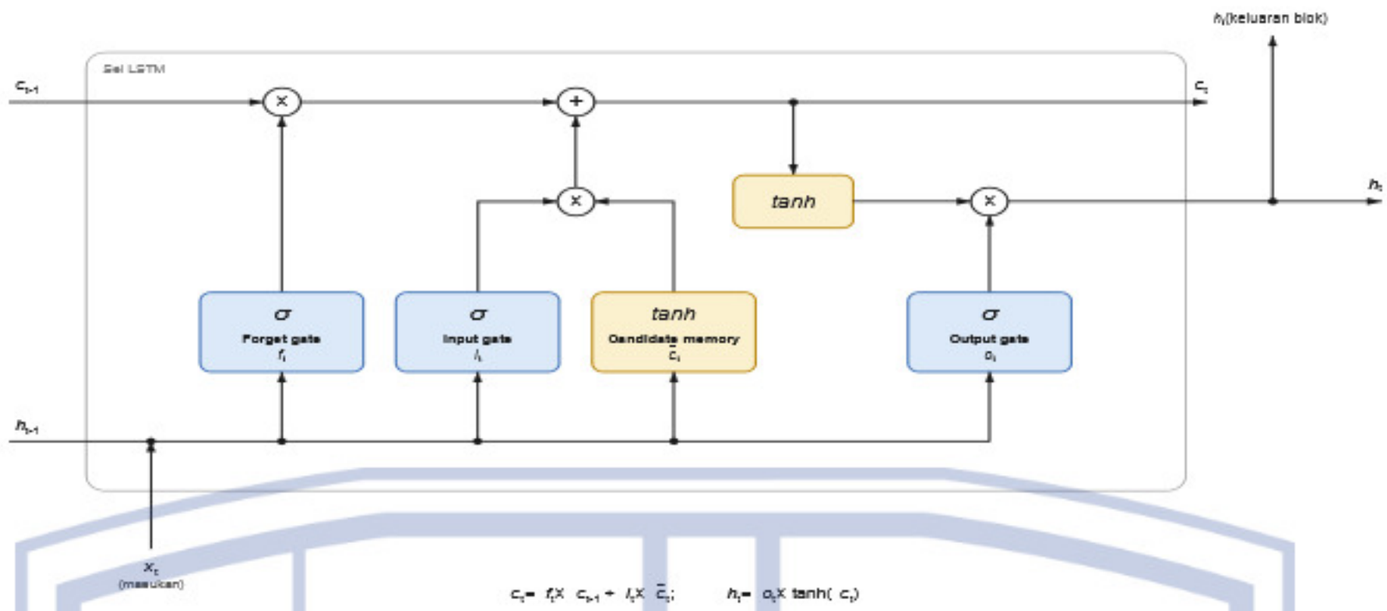
2.5.1 Konsep dan Cara Kerja LSTM

Long Short-Term Memory adalah jenis jaringan saraf yang dirancang khusus untuk data berurutan seperti deret waktu [19]. Berbeda dengan jaringan biasa yang mengolah setiap masukan secara terpisah, jaringan jenis ini mengolah data satu per satu menurut urutannya sambil membawa ingatan dari langkah sebelumnya. Ingatan tersebut memungkinkan model menghubungkan kejadian hari ini dengan kejadian beberapa hari atau bahkan beberapa bulan sebelumnya. Jaringan berulang sederhana sebenarnya sudah memiliki ingatan, tetapi ingatannya cepat memudar ketika urutannya panjang. Masalah memudarnya ingatan itu terjadi karena sinyal perbaikan yang dikirim mundur saat pelatihan menyusut sampai hampir hilang. LSTM mengatasi masalah tersebut dengan menambahkan jalur ingatan khusus yang dapat mempertahankan informasi dalam waktu lama. Karena itu LSTM banyak dipakai pada peramalan yang membutuhkan ingatan panjang.

2.5.2 Struktur Sel dan Gerbang LSTM

Bagian utama LSTM adalah sel ingatan yang berfungsi seperti wadah penyimpan informasi. Isi wadah tersebut diatur oleh tiga pintu pengatur yang menentukan informasi mana yang boleh masuk, bertahan, atau keluar. Pintu pelupa memutuskan bagian mana dari ingatan lama yang sebaiknya dibuang karena sudah tidak berguna. Pintu masukan menentukan seberapa banyak informasi baru dari data hari ini yang layak disimpan ke dalam ingatan. Pintu keluaran mengatur bagian mana dari ingatan yang dipakai untuk menghasilkan jawaban pada langkah tersebut. Setiap pintu memakai fungsi pengaktif yang menghasilkan angka antara nol dan satu, sehingga dapat diartikan sebagai pintu yang tertutup rapat, terbuka sebagian, atau terbuka penuh. Melalui tiga pintu inilah LSTM dapat memilih sendiri informasi mana yang perlu diingat lama dan mana yang boleh segera dilupakan. Struktur sel ingatan dan ketiga pintu pengatur yang membentuk mekanisme inti LSTM dapat

dilihat pada Gambar 2.5, sedangkan ketiga persamaan pintunya dinyatakan berturut-turut pada Persamaan (2.2), (2.3), dan (2.4).



Gambar 2.5 Struktur Sel Ingatan dan Pintu Pengatur pada LSTM [19]

$$f_t = \sigma\left(W_f[h_{t-1}, x_t] + b_f\right) \quad (2.2)$$

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \quad (2.3)$$

$$\tilde{c}_t = \tanh(W_c[h_{t-1}, x_t] + b_c) \quad (2.4)$$

Dimana:

f = pintu pelupa,

i = pintu masukan,

o = pintu keluaran,

W = bobot,

b = bias,

h = keluaran langkah sebelumnya, dan

x = masukan saat ini.

Ketiga persamaan di atas bekerja dengan logika yang sama: nilai masukan saat ini dan keluaran langkah sebelumnya digabungkan, lalu dilewatkan pada fungsi sigmoid yang menghasilkan angka antara nol dan satu. Angka tersebut berfungsi seperti katup yang mengatur seberapa besar bagian informasi yang diizinkan mengalir melalui masing-masing pintu.

Ketiga pintu tersebut dihitung dengan cara yang serupa, yaitu menggabungkan masukan hari ini dengan keluaran langkah sebelumnya, lalu melewatkannya pada fungsi pengaktif. Hasil perhitungan pintu kemudian dipakai untuk memperbarui isi sel ingatan. Ingatan lama dikalikan dengan nilai pintu pelupa, sehingga bagian yang dianggap tidak berguna akan mengecil mendekati nol. Informasi baru dikalikan dengan nilai pintu masukan, lalu ditambahkan ke ingatan yang telah disaring tersebut. Keluaran langkah itu diperoleh dengan melewatkan isi ingatan pada fungsi pengaktif dan mengalikannya dengan nilai pintu

keluaran. Rangkaian perhitungan ini diulang untuk setiap langkah waktu sepanjang deret data. Dengan cara demikian, informasi penting dapat mengalir jauh ke depan tanpa cepat memudar. Cara memperbarui sel ingatan dan menghasilkan keluaran tersembunyi dinyatakan pada Persamaan (2.5) dan (2.6).

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad \text{.....(2.5)}$$

$$h_t = o_t \odot \tanh(c_t) \quad \text{.....(2.6)}$$

Dimana:

C = isi sel ingatan,

C dengan tanda gelombang = calon ingatan baru, dan

h = keluaran tersembunyi.

Persamaan (2.5) menunjukkan bahwa ingatan baru terbentuk dari campuran ingatan lama yang disaring oleh pintu pelupa dan informasi baru yang disaring oleh pintu masukan. Persamaan (2.6) kemudian mengubah isi ingatan tersebut menjadi keluaran tersembunyi yang akan diteruskan ke langkah berikutnya.

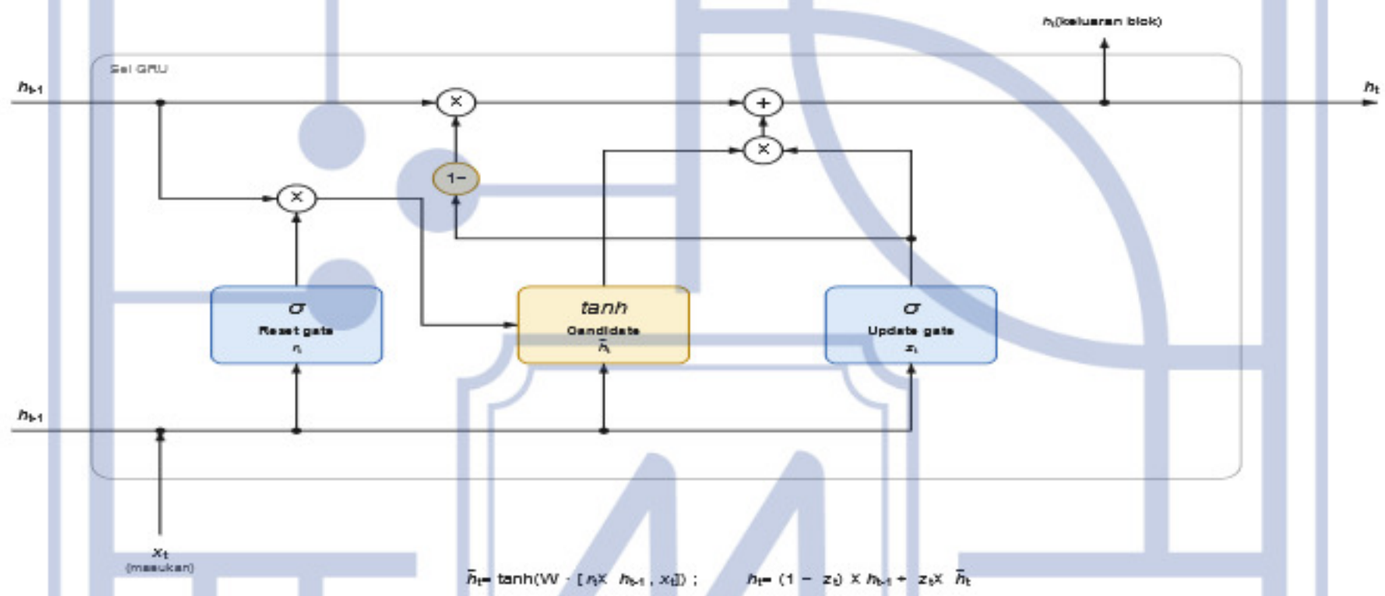
2.6 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) adalah arsitektur jaringan berulang yang dikembangkan sebagai penyederhanaan dari LSTM dengan tetap mempertahankan mekanisme gerbang untuk mengelola aliran informasi. Secara umum, GRU hanya memiliki dua gerbang, yaitu gerbang pembaru dan gerbang penyetel ulang, sehingga jumlah parameternya lebih sedikit dibandingkan LSTM yang menggunakan tiga gerbang dengan sel memori terpisah. Desain yang lebih ringkas ini membuat GRU umumnya lebih cepat dilatih dengan tetap mampu menangkap ketergantungan jangka menengah hingga panjang pada data deret waktu. Ilustrasi struktur sel GRU dengan kedua gerbangnya secara konseptual dapat dilihat pada Gambar 2.6. Perbandingan antara GRU dan LSTM dalam penelitian terdahulu menunjukkan hasil yang tidak seragam, sehingga keduanya perlu diuji secara empiris pada data yang sama. Subbab berikut menguraikan mekanisme gerbang GRU serta perbedaan mendasarnya dengan LSTM.

2.6.1 Konsep dan Cara Kerja GRU

Gated Recurrent Unit adalah jenis jaringan berurutan yang dikembangkan sesudah LSTM dengan tujuan menyederhanakan susunannya [20]. Gagasan dasarnya sama, yaitu memakai pintu pengatur agar informasi penting dapat bertahan lama dan informasi yang tidak berguna dapat dibuang. Perbedaannya, GRU hanya memakai dua pintu dan tidak

memisahkan wadah ingatan dari keluaran. Pintu pembaru menentukan seberapa banyak informasi lama yang dipertahankan dibandingkan informasi baru. Pintu penyetel ulang menentukan seberapa besar pengaruh keluaran sebelumnya ketika menghitung calon jawaban baru. Karena pintunya lebih sedikit, jumlah bobot yang harus dipelajari GRU lebih kecil daripada LSTM. Susunan yang lebih ringkas ini membuat GRU biasanya lebih cepat dilatih pada jumlah data yang sama. Struktur sel GRU dengan pintu pembaru dan pintu penyetel ulang ditunjukkan secara umum pada Gambar 2.6, sedangkan operasi matematis kedua gerbang beserta calon keluaran barunya dinyatakan pada Persamaan (2.7a), (2.7b), dan (2.7c).



Gambar 2.6 Struktur Sel GRU dengan Pintu Pembaru dan Pintu Penyetel Ulang [20]

$$z_t = \sigma(W_{zh} \cdot h_{t-1} + W_{zx} \cdot x_t + b_z) \quad \text{(2.7a)}$$

$$r_t = \sigma(W_{rh} \cdot h_{t-1} + W_{rx} \cdot x_t + b_r) \quad \text{(2.7b)}$$

$$\tilde{h}_t = \tanh(W_{hh} \cdot (r_t \odot h_{t-1}) + W_{hx} \cdot x_t + b_h) \quad \text{(2.7c)}$$

Dimana:

z = pintu pembaru,

r = pintu penyetel ulang, h dengan tanda gelombang = calon keluaran baru, dan

h = keluaran akhir langkah tersebut.

Persamaan (2.7a) menghitung nilai gerbang pembaru yang menentukan seberapa banyak informasi dari langkah sebelumnya dipertahankan. Nilai mendekati satu berarti informasi lama lebih banyak dipertahankan. Persamaan (2.7b) menghitung gerbang penyetel ulang yang mengontrol seberapa besar pengaruh keluaran sebelumnya saat membentuk calon keluaran baru, dan Persamaan (2.7c) menghitung calon keluaran baru dari kombinasi informasi yang telah disaring tersebut.

Keluaran akhir satu langkah GRU dinyatakan pada Persamaan (2.7d).

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad \text{(2.7d)}$$

Dimana:

h dengan indeks t = keluaran akhir langkah tersebut,

z dengan indeks t = gerbang pembaru dari Persamaan (2.7a),

h dengan tanda gelombang indeks t = calon keluaran baru dari Persamaan (2.7c), dan tanda $\odot$ = perkalian unsur demi unsur.

Persamaan ini memperlihatkan bahwa GRU mencampur keluaran langkah sebelumnya dengan calon keluaran baru memakai satu gerbang pembaru saja, tanpa sel ingatan terpisah sebagaimana pada Persamaan (2.5). Inilah sebab GRU memiliki bobot lebih sedikit daripada LSTM.

2.6.2 Perbedaan GRU dengan LSTM

Perbedaan pertama antara GRU dan LSTM terletak pada jumlah pintu pengaturnya, yaitu dua berbanding tiga. Perbedaan kedua terletak pada cara menyimpan ingatan, sebab LSTM memiliki wadah ingatan terpisah sedangkan GRU menyatukannya dengan keluaran. Akibat kedua perbedaan itu, GRU memiliki bobot yang lebih sedikit sehingga pelatihannya cenderung lebih ringan. Susunan yang lebih sederhana kadang menguntungkan ketika jumlah data latih terbatas, karena model dengan bobot lebih sedikit lebih kecil kemungkinannya menghafal. Sebaliknya, wadah ingatan terpisah pada LSTM dapat menguntungkan ketika pola yang harus diingat sangat panjang. Penelitian terdahulu belum sepakat mengenai arsitektur mana yang selalu lebih unggul, sebab hasilnya bergantung pada data dan cara pengujiannya [10]. Karena itu penelitian ini menguji ketiganya pada data dan pengaturan yang sama agar perbandingannya adil.

2.6.3 Perhitungan Jumlah Parameter CNN, LSTM, dan GRU

Jumlah parameter yang dapat dilatih pada ketiga arsitektur di atas dapat dihitung terlebih dahulu secara analitis dari rumus umum lapisannya, sebelum angka final untuk model yang benar-benar dipakai penelitian ini ditampilkan pada Bab 3 dan Bab 4. Rumus umum tersebut adalah sebagai berikut:

Jumlah parameter lapisan konvolusi dinyatakan pada Persamaan (2.1b).

$$P_{conv} = (k \times C_{in} \times F) + F \quad \text{.....(2.1b)}$$

Dimana:

k = panjang penapis, C dengan indeks in = jumlah saluran masukan, dan F = jumlah penapis.

Jumlah parameter lapisan padat dinyatakan pada Persamaan (2.1c).

$$P_{dense} = (n_{in} \times n_{out}) + n_{out} \quad \text{.....(2.1c)}$$

Dimana:

n dengan indeks in = jumlah masukan, dan n dengan indeks out = jumlah keluaran lapisan tersebut.

Jumlah parameter lapisan LSTM dinyatakan pada Persamaan (2.6a).

$$P_{lstm} = 4 \times [(m + u) \times u + u] \quad \text{.....(2.6a)}$$

Dimana:

m = jumlah fitur masukan, u = jumlah unit, dan angka 4 berasal dari empat kelompok bobot pada Persamaan (2.2), (2.3), (2.4), dan gerbang keluaran.

Jumlah parameter lapisan GRU dinyatakan pada Persamaan (2.7e).

$$P_{gru} = 3 \times [(m + u) \times u] + (3 \times 2 \times u) \quad \text{.....(2.7e)}$$

Dimana:

angka 3 berasal dari tiga kelompok bobot pada Persamaan (2.7a), (2.7b), dan (2.7c), sedangkan suku $3 \times 2 \times u$ muncul karena pustaka Keras memakai penyetelan reset_after = True sehingga bias dihitung dua kali, yaitu satu untuk jalur masukan dan satu untuk jalur berulang.

Rumus-rumus di atas menjelaskan mengapa, pada jumlah unit yang sama, LSTM selalu memiliki parameter paling banyak, diikuti GRU, kemudian CNN—konsisten dengan urutan besaran parameter yang ditampilkan pada Tabel 4.2. Angka final tiap model (jumlah unit, jumlah filter, dan jumlah fitur masukan yang benar-benar dipakai) dijelaskan pada Bab 3 dan Bab 4.

Karena lapisan keluaran pada penelitian ini berukuran enam agar keenam kolom diprediksi sekaligus, angka jumlah parameter yang berlaku adalah 36.166 untuk CNN, 70.356 untuk LSTM, dan 53.331 untuk GRU, sebagaimana dihitung pada Subbab 3.2.0 proses ketujuh.

2.7 Pra-pemrosesan Data

Pra-pemrosesan data adalah serangkaian langkah transformasi yang diterapkan pada data mentah sebelum dimasukkan ke model pembelajaran mendalam, bertujuan untuk meningkatkan kualitas data dan kesesuaiannya dengan asumsi model. Secara umum, tahapan pra-pemrosesan dalam penelitian ini mencakup tiga langkah utama yang dijalankan secara berurutan, yaitu transformasi log-return untuk menstabilkan deret harga yang tidak stasioner, penskalaan Min-Max untuk menyamakan rentang nilai antarfitur, dan pembentukan jendela geser untuk mengubah deret satu dimensi menjadi pasangan masukan-keluaran yang dapat dipelajari model. Mekanisme jendela geser yang digunakan dalam penelitian ini secara

konseptual dapat dilihat pada Gambar 2.7. Urutan ketiga langkah ini penting untuk dicermati karena pengubahan urutan dapat menyebabkan kebocoran data atau menghasilkan skala yang tidak konsisten antara data latih dan data uji. Parameter transformasi, termasuk nilai minimum dan maksimum untuk penskalaan, selalu dihitung hanya dari data latih pada setiap *fold* validasi untuk mencegah kebocoran informasi dari data uji. Subbab-subbab berikut membahas setiap langkah pra-pemrosesan tersebut secara lebih rinci beserta landasan pemilihannya.

2.7.1 Transformasi Log-Return

Log-return adalah cara mengubah deret harga menjadi deret perubahan harian dalam bentuk logaritma. Nilainya diperoleh dengan membagi harga hari ini dengan harga hari sebelumnya, lalu mengambil logaritma dari hasil pembagian tersebut. Pengubahan ini dilakukan karena deret harga mentah tidak stabil dan tumbuh berkali-kali lipat sepanjang periode data. Setelah diubah, deret yang dihasilkan berkisar di sekitar nol dan sifatnya jauh lebih stabil [15]. Bentuk logaritma juga memiliki sifat praktis, yaitu perubahan beberapa hari berturut-turut dapat dijumlahkan secara langsung. Sifat tersebut memudahkan penyusunan kembali harga dari deret perubahan ketika hasil ramalan hendak dibaca dalam satuan mata uang. Dengan demikian model belajar dari pola perubahan harga, bukan dari tingkat harganya yang terus menanjak. Rumus perhitungan log-return dinyatakan pada Persamaan (2.8).

$$r_t = \ln\left(\frac{P_t}{P_{t-1}}\right) \quad \text{.....(2.8)}$$

Dimana:

r = log-return pada hari ke-t,

P = harga pada hari tersebut, dan

ln = logaritma alami.

Rumus ini pada dasarnya mengukur seberapa besar harga berubah dari hari sebelumnya dalam skala logaritma. Nilai positif berarti harga naik dan nilai negatif berarti harga turun. Karena logaritma bersifat simetris, kenaikan dan penurunan dengan persentase yang sama menghasilkan nilai log-return yang sama besar namun berlawanan tanda.

Penyusunan kembali harga dari log-return dinyatakan pada Persamaan (2.8a).

$$P_t = P_{t-1} \times \exp(r_t) \quad \text{.....(2.8a)}$$

Dimana:

P dengan indeks t = harga pada hari ke-t, P dengan indeks t-1 = harga hari sebelumnya, dan r dengan indeks t = log-return dari Persamaan (2.8).

Persamaan ini adalah kebalikan Persamaan (2.8), sehingga hasil ramalan yang berupa perubahan harian dapat dibaca kembali dalam satuan dolar.

2.7.2 Penskalaan Data dengan Min-Max

Penskalaan adalah proses menyamakan rentang nilai antarfitur agar tidak ada fitur yang mendominasi pembelajaran. Penyamaan ini diperlukan karena fitur yang dipakai memiliki satuan dan besaran yang sangat berbeda, misalnya jumlah lembar saham yang diperdagangkan bernilai jutaan sedangkan harga hanya puluhan hingga ratusan. Tanpa penyamaan, fitur berangka besar akan dianggap lebih penting oleh model semata-mata karena angkanya besar. Penskalaan Min-Max bekerja dengan mengubah nilai terkecil menjadi nol dan nilai terbesar menjadi satu, sedangkan nilai lain diletakkan di antaranya secara sebanding. Nilai terkecil dan terbesar yang dipakai hanya diambil dari data latih, bukan dari seluruh data. Pembatasan tersebut penting untuk mencegah kebocoran data yang dijelaskan pada Subbab 2.8.2. Akibat pembatasan itu, nilai pada data uji dapat sedikit berada di luar rentang nol sampai satu apabila melampaui rentang data latihnya. Rumus penskalaan Min-Max dinyatakan pada Persamaan (2.9).

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad \text{.....(2.9)}$$

Dimana:

x aksen = nilai setelah diskalakan,

x = nilai asli, x minimum dan

x maksimum = nilai terkecil dan terbesar pada data latih.

Rumus ini menggeser dan meregangkan nilai asli sehingga nilai terkecil menjadi tepat nol dan nilai terbesar menjadi tepat satu, sementara seluruh nilai di antaranya tetap terdistribusi secara sebanding. Dengan cara ini semua fitur berada dalam rentang yang sama tanpa mengubah bentuk distribusinya.

Pengembalian nilai dari skala normalisasi ke skala asli dinyatakan pada Persamaan (2.9a).

$$x = x' \times (x_{max} - x_{min}) + x_{min} \quad \text{.....(2.9a)}$$

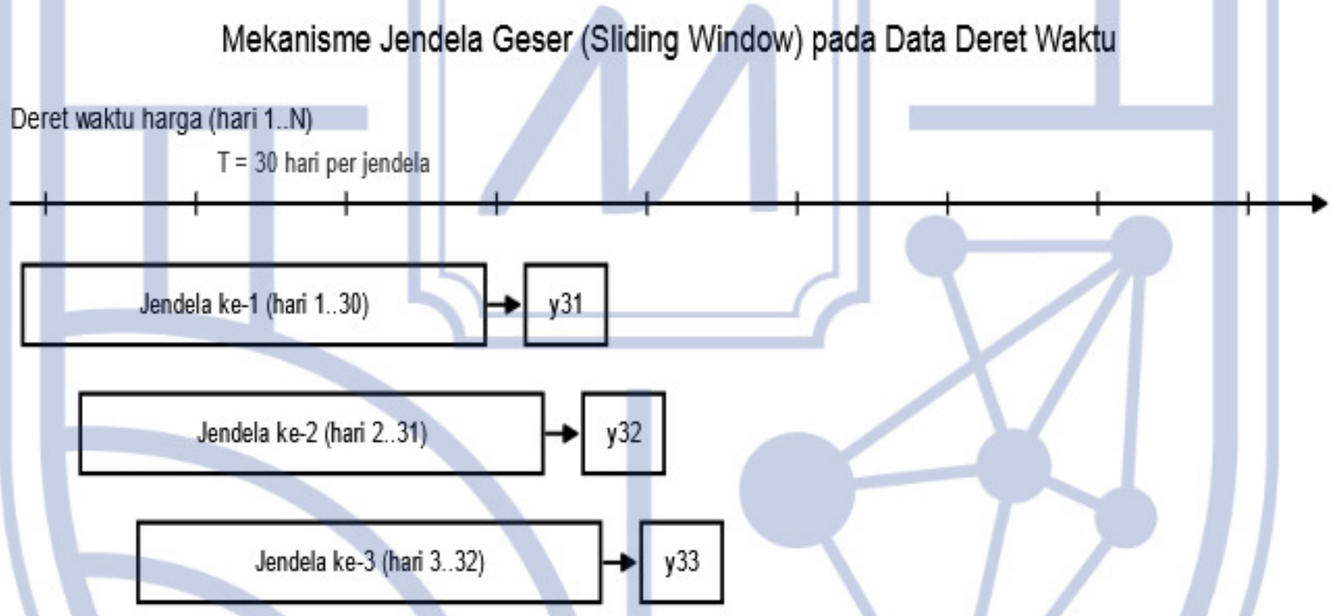
Dimana:

x aksen = nilai pada skala normalisasi, dan x minimum serta

x maksimum = nilai terkecil dan terbesar yang dihitung hanya dari data latih.

Persamaan ini adalah kebalikan Persamaan (2.9) dan diperlukan karena keluaran model masih berada pada skala normalisasi.

Selain kedua pengubahan di atas, data juga disusun menjadi jendela geser sebelum dimasukkan ke model. Jendela geser berarti mengambil sejumlah hari berurutan sebagai masukan, lalu meminta model memperkirakan nilai hari berikutnya. Jendela tersebut kemudian digeser satu hari ke depan, dan proses yang sama diulang sampai akhir data. Cara ini mengubah satu deret panjang menjadi banyak pasangan masukan dan jawaban yang dapat dipelajari model. Panjang jendela menentukan seberapa jauh ke belakang model boleh melihat ketika membuat ramalan. Jendela yang terlalu pendek membuat model kekurangan konteks, sedangkan jendela yang terlalu panjang membuat perhitungan berat dan menambah risiko menghafal. Panjang jendela yang dipakai pada penelitian ini ditetapkan melalui percobaan yang diuraikan pada Bab III. Mekanisme jendela geser yang mengubah deret satu dimensi menjadi pasangan masukan-keluaran dapat dilihat pada Gambar 2.7.



Gambar 2.7 Mekanisme Jendela Geser pada Data Deret Waktu [15]

2.7.3 Deteksi Nilai Ekstrem dengan Metode IQR Tukey

Nilai ekstrem adalah pengamatan yang terletak jauh dari sebaran utama data. Metode kuartil Tukey menandai nilai ekstrem berdasarkan jarak terhadap kuartil pertama dan ketiga. Rentang interkuartil dinyatakan pada Persamaan (2.9b).

$$\text{IQR} = Q_3 - Q_1 \quad \text{.....(2.9b)}$$

Batas bawah dan batas atas kewajaran dinyatakan pada Persamaan (2.9c) dan (2.9d).

$$\text{Batas bawah} = Q_1 - (1{,}5 \times \text{IQR}) \quad \text{.....(2.9c)}$$

$$\text{Batas atas} = Q_3 + (1{,}5 \times \text{IQR}) \quad \text{.....(2.9d)}$$

Dimana:

Q dengan indeks 1 = kuartil pertama, yaitu nilai yang membatasi dua puluh lima persen data terkecil,

Q dengan indeks 3 = kuartil ketiga, yaitu nilai yang membatasi dua puluh lima persen data terbesar, dan angka 1,5 = pengali lazim yang diperkenalkan Tukey sebagai batas kewajaran.

Pada data harga saham berjangka panjang, nilai yang tertandai ekstrem belum tentu kesalahan pencatatan, sebab harga yang tumbuh berkali-kali lipat membuat periode akhir otomatis terbaca ekstrem terhadap keseluruhan sebaran. Paragraf ini menjadi dasar keputusan pada BAB III untuk mempertahankan seluruh baris.

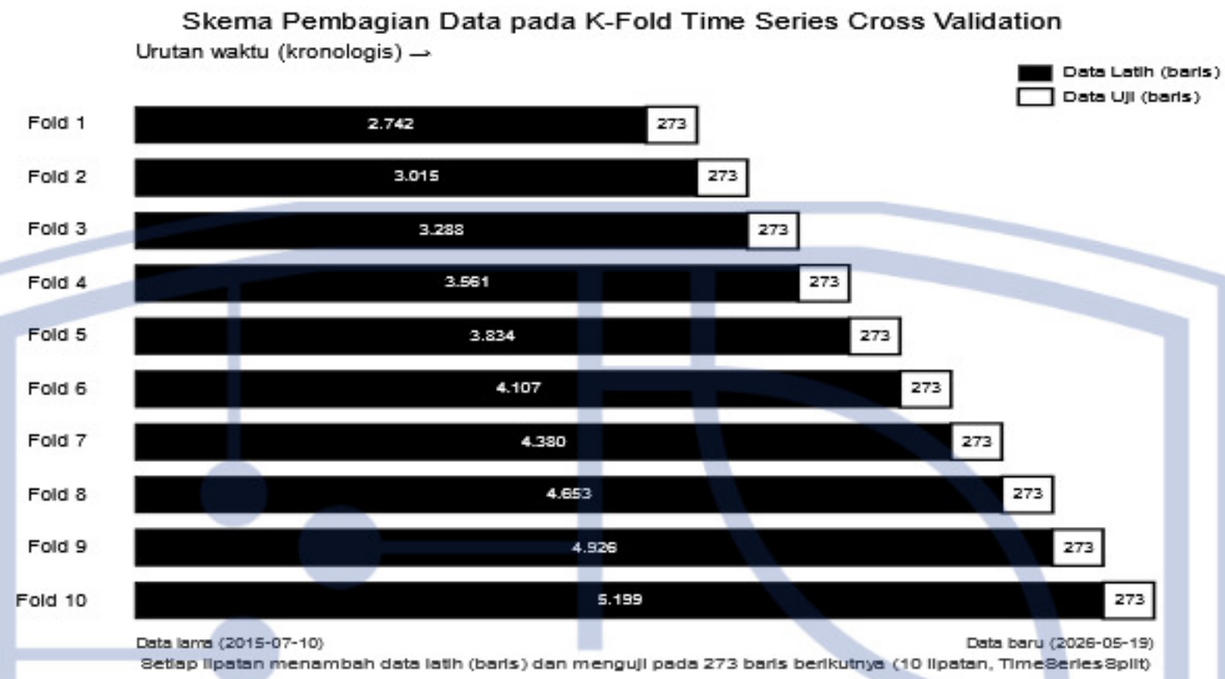
2.8 K-Fold Time Series Cross Validation

K-Fold Time Series Cross Validation adalah kerangka evaluasi model yang menggabungkan prinsip validasi silang dengan penghormatan terhadap urutan kronologis data deret waktu. Secara umum, skema ini membagi data menjadi sejumlah K lipatan yang berurutan sehingga data uji pada setiap *fold* selalu berada di masa yang lebih baru daripada data latihnya, menghindari kemungkinan model mempelajari informasi masa depan secara tidak sah. Perbedaan mendasar ini membedakan K-Fold Time Series dari K-Fold biasa yang membagi data secara acak tanpa mempertimbangkan urutan waktu. Skema pembagian data pada K-Fold Time Series Cross Validation yang dipakai dalam penelitian ini secara konseptual dapat dilihat pada Gambar 2.8. Dalam penelitian ini digunakan 10 lipatan dengan pendekatan TimeSeriesSplit dari pustaka scikit-learn, sehingga setiap model dilatih dan diuji sebanyak sepuluh kali pada segmen data yang berbeda. Subbab-subbab berikut menguraikan perbedaannya dengan K-Fold biasa serta mekanisme pencegahan kebocoran data yang diterapkan.

2.8.1 Perbedaan dengan K-Fold Biasa

Validasi silang adalah cara menguji model dengan membagi data menjadi beberapa bagian, lalu melatih dan menguji model berulang kali pada bagian yang berbeda-beda. Tujuannya agar penilaian tidak bergantung pada satu pembagian data yang mungkin kebetulan menguntungkan. Pada K-Fold biasa, data dibagi secara acak sehingga bagian mana pun dapat menjadi data uji. Cara acak tersebut tidak sesuai untuk deret waktu, sebab dapat terjadi model dilatih memakai data hari yang lebih baru lalu diuji pada hari yang lebih lama. Keadaan itu sama saja dengan membiarkan model mengetahui masa depan sebelum diuji, sehingga hasilnya tampak jauh lebih baik daripada keadaan sebenarnya [14]. K-Fold

untuk deret waktu memperbaiki hal ini dengan selalu menempatkan bagian uji sesudah bagian latih menurut urutan waktu. Dengan begitu penilaian yang diperoleh mencerminkan kemampuan model meramal hari yang benar-benar belum terjadi. Skema pembagian data pada K-Fold Time Series Cross Validation yang dipakai dalam penelitian ini dapat dilihat pada Gambar 2.8.



Gambar 2.8 Skema Pembagian Data pada K-Fold Time Series Cross Validation [14]

Pada skema ini, pengujian dilakukan bertahap sebanyak sepuluh putaran. Putaran pertama memakai potongan data paling awal sebagai data latih dan potongan sesudahnya sebagai data uji. Pada putaran berikutnya, potongan yang tadinya menjadi data uji digabungkan ke data latih, lalu potongan sesudahnya dipakai sebagai data uji yang baru. Proses tersebut berulang sehingga data latih terus bertambah panjang dan data uji selalu bergerak maju. Setiap putaran menghasilkan satu nilai penilaian, dan nilai dari seluruh putaran kemudian dirata-ratakan. Rata-rata tersebut lebih dapat dipercaya daripada hasil satu kali pengujian saja, karena sudah mencakup berbagai periode pasar. Simpangan antarputaran juga memberi gambaran seberapa stabil kemampuan model pada kondisi pasar yang berbeda-beda.

2.8.2 Pencegahan Kebocoran Data (Data Leakage)

Kebocoran data adalah keadaan ketika informasi yang seharusnya belum diketahui model ikut terpakai selama pelatihan. Pada peramalan deret waktu, kebocoran paling sering terjadi karena urutan waktu tidak dijaga saat membagi data. Bentuk kebocoran lain yang mudah terlewat adalah penghitungan nilai penskalaan memakai seluruh data, termasuk data uji. Apabila nilai terkecil dan terbesar dihitung dari seluruh data, maka model secara tidak langsung sudah mengetahui rentang nilai masa depan sebelum diuji [14]. Akibat kebocoran, hasil pengujian terlihat bagus di atas kertas tetapi tidak dapat diulang pada keadaan nyata.

Untuk mencegahnya, penelitian ini menghitung nilai penskalaan hanya dari data latih pada setiap putaran. Urutan waktu juga selalu dijaga sehingga data uji berada sesudah data latih. Kedua langkah tersebut membuat angka yang dilaporkan mencerminkan kemampuan model yang sesungguhnya.

2.9 Evaluasi Model dan Peramalan Dasar

Menilai sebuah model peramalan tidak cukup memakai satu ukuran saja, sebab setiap ukuran menyoroti sisi yang berbeda. Ukuran yang satu dapat memperlihatkan seberapa jauh ramalan meleset dalam satuan harga, sedangkan ukuran lain memperlihatkan apakah model berhasil memperkirakan arah naik atau turun. Sebuah model bisa saja terlihat sangat baik pada satu ukuran namun ternyata lemah pada ukuran lainnya [15]. Karena itu penelitian ini memakai empat ukuran yang saling melengkapi. Keempat ukuran tersebut adalah rata-rata kesalahan mutlak, koefisien penentu, ketepatan arah, dan kesalahan terskala terhadap peramalan dasar. Selain keempatnya, hasil model juga dibandingkan dengan peramalan dasar sebagai pembanding paling dasar. Perbandingan terhadap pembanding dasar diperlukan agar dapat diketahui apakah model benar-benar memberikan manfaat tambahan.

2.9.1 Mean Absolute Error (MAE)

Rata-rata kesalahan mutlak mengukur seberapa jauh ramalan model meleset dari nilai sebenarnya, tanpa memandang arah melesetnya. Cara menghitungnya adalah mengambil selisih antara nilai ramalan dan nilai sebenarnya pada setiap hari, membuang tanda positif atau negatifnya, lalu merata-ratakan seluruh selisih tersebut. Nilainya dinyatakan dalam satuan yang sama dengan data aslinya, sehingga mudah dibaca. Apabila ukuran ini bernilai dua dolar, artinya ramalan model rata-rata meleset sekitar dua dolar dari harga yang sebenarnya. Semakin kecil nilainya, semakin baik kemampuan model, dan nilai nol berarti ramalan tepat sempurna. Ukuran ini mudah dipahami tetapi tidak memberi tahu apakah model berhasil memperkirakan arah pergerakan. Karena itu ukuran ini perlu dibaca bersama ukuran lain. Rumus perhitungan MAE dinyatakan pada Persamaan (2.10).

$$\text{MAE} = \frac{1}{n}\sum_i |y_i - \hat{y}_i| \quad \text{.....(2.10)}$$

Dimana:

n = banyaknya data,

y = nilai sebenarnya, dan

y topi = nilai ramalan.

Rumus ini menjumlahkan seluruh nilai selisih antara ramalan dan nilai sebenarnya tanpa memandang apakah selisihnya positif atau negatif, lalu membaginya dengan banyaknya data untuk mendapatkan rata-rata. Hasilnya dinyatakan dalam satuan yang sama dengan data aslinya sehingga nilainya langsung dapat diinterpretasikan sebagai rata-rata penyimpangan ramalan.

Fungsi rugi kesalahan kuadrat rata-rata dinyatakan pada Persamaan (2.10a).

$$\text{MSE} = \frac{1}{n} \times \sum_i (y_i - \hat{y}_i)^2 \quad \text{.....(2.10a)}$$

Dimana:

n = banyaknya data,

y dengan indeks i = nilai sebenarnya, dan

y topi dengan indeks i = nilai ramalan.

Berbeda dengan MAE pada Persamaan (2.10) yang membuang tanda selisih, MSE mengkuadratkan selisih sehingga kesalahan besar dihukum lebih berat. Sifat itulah yang membuat MSE dipakai sebagai fungsi rugi selama pelatihan, sedangkan MAE dipakai sebagai ukuran pelaporan karena satuannya sama dengan data asli.

2.9.2 Koefisien Determinasi (R Kuadrat)

Koefisien penentu mengukur seberapa besar bagian dari naik-turunnya data yang berhasil dijelaskan oleh model. Nilainya berkisar sampai satu, dan semakin mendekati satu berarti garis ramalan semakin berimpit dengan garis data sebenarnya. Nilai nol berarti model tidak lebih baik daripada sekadar memperkirakan dengan nilai rata-rata. Meskipun demikian, ukuran ini perlu dibaca dengan sangat hati-hati pada data harga saham. Harga saham hari ini selalu berdekatan dengan harga kemarin, sehingga ramalan yang sekadar menyalin harga kemarin pun sudah menghasilkan nilai yang sangat tinggi. Dengan kata lain, nilai yang tinggi belum tentu berarti model mampu memperkirakan arah pergerakan harga. Oleh karena itu ukuran ini tidak boleh dijadikan satu-satunya bukti keberhasilan sebuah model peramalan harga saham. Rumus koefisien determinasi dinyatakan pada Persamaan (2.11).

$$R^2 = 1 - \left(\frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}\right) \quad \text{.....(2.11)}$$

Dimana:

y = nilai sebenarnya,

y topi = nilai ramalan, dan

y garis = rata-rata nilai sebenarnya.

Rumus ini membandingkan jumlah kesalahan kuadrat model dengan jumlah ragam total data. Semakin kecil kesalahan model dibandingkan ragam total, semakin besar nilai R^2 yang mendekati satu. Nilai $R^2 = 1$ berarti model menjelaskan seluruh variasi data secara sempurna, sedangkan nilai $R^2 = 0$ berarti model tidak lebih baik daripada hanya menggunakan nilai rata-rata sebagai ramalan.

2.9.3 Ketepatan Arah dan Analogi Lemparan Koin

Ketepatan arah mengukur seberapa sering model berhasil memperkirakan dengan benar apakah harga akan naik atau turun pada hari berikutnya. Ukuran ini tidak memperhatikan besar kecilnya selisih, melainkan hanya benar atau salahnya arah perkiraan. Nilainya dinyatakan dalam persen, yaitu banyaknya perkiraan arah yang benar dibagi seluruh perkiraan. Untuk memahami maknanya, bayangkan sebuah koin yang dilempar ke udara. Koin memiliki dua sisi dengan peluang yang sama, sehingga memperkirakan sisi mana yang muncul akan benar kira-kira setengah dari seluruh lemparan. Begitu pula memperkirakan arah harga saham secara acak akan benar sekitar lima puluh persen dari waktu. Angka lima puluh persen inilah yang menjadi batas pembanding paling dasar bagi ukuran ketepatan arah.

Penafsirannya harus dinyatakan dengan tegas agar tidak menyesatkan. Apabila ketepatan arah sebuah model tepat lima puluh persen, artinya model tidak lebih pandai daripada memperkirakan secara acak, atau setara dengan lemparan koin. Apabila nilainya berada dibawah lima puluh persen, artinya model justru lebih buruk daripada perkiraan acak, sebab memperkirakan kebalikan dari jawaban model akan lebih sering benar. Hanya apabila nilainya berada di atas lima puluh persen barulah model dapat dikatakan memiliki sedikit kemampuan memperkirakan arah. Bahkan pada keadaan itu pun, selisih beberapa persen di atas lima puluh masih perlu diuji secara statistik untuk memastikan bahwa selisih tersebut bukan sekadar kebetulan. Uji yang dipakai untuk keperluan tersebut diuraikan pada Bab III. Dengan cara pembacaan seperti ini, ukuran ketepatan arah menjadi pembanding yang jujur terhadap perkiraan acak. Rumus perhitungan ketepatan arah dinyatakan pada Persamaan (2.12).

$$\mathrm{DA} = \frac{100\%}{m} \sum_i I\left(\mathrm{sign}(\Delta y_i) = \mathrm{sign}(\Delta \hat{y}_i)\right) \quad \text{.....(2.12)}$$

Dimana:

n = banyaknya data, dan

fungsi tanda bernilai satu apabila arah ramalan sama dengan arah sebenarnya.

Rumus ini menghitung persentase kejadian di mana tanda arah ramalan (naik atau turun) sama dengan tanda arah nilai sebenarnya. Fungsi indikator bernilai satu jika arahnya cocok dan nol jika tidak. Nilai DA sebesar 50% menunjukkan kinerja yang setara dengan perkiraan acak, sehingga nilai yang bermakna harus secara konsisten berada di atas ambang ini.

2.9.4 Peramalan Dasar sebagai Pembanding dan MASE

Peramalan dasar (naive forecast) adalah cara meramal paling sederhana, yaitu memperkirakan bahwa harga besok akan sama persis dengan harga hari ini. Cara ini tidak memerlukan perhitungan rumit maupun pelatihan model, tetapi pada data harga saham hasilnya sering kali sudah cukup baik. Kenyataan tersebut membuat peramalan dasar layak dijadikan pembanding paling dasar bagi model yang rumit [15]. Sebuah model pembelajaran mendalam baru dapat dikatakan bermanfaat apabila kesalahannya lebih kecil daripada kesalahan peramalan dasar. Apabila kesalahannya justru lebih besar, maka seluruh kerumitan model tersebut tidak memberi keuntungan apa pun. Untuk membandingkan keduanya secara langsung digunakan ukuran kesalahan terskala. Ukuran tersebut membagi kesalahan model dengan kesalahan peramalan dasar pada data latih.

Penafsiran ukuran kesalahan terskala sangat sederhana karena hasilnya berupa perbandingan. Apabila nilainya tepat satu, artinya kesalahan model sama besar dengan kesalahan perkiraan dasar, sehingga model tidak memberi manfaat tambahan. Apabila nilainya lebih besar daripada satu, misalnya satu koma nol dua, artinya kesalahan model dua persen lebih buruk daripada perkiraan dasar, dan keadaan ini harus dinilai sebagai hasil yang buruk. Sebaliknya, apabila nilainya lebih kecil daripada satu, misalnya nol koma sembilan, artinya kesalahan model sepuluh persen lebih kecil daripada perkiraan dasar, sehingga model dapat dikatakan berhasil mengalahkan pembanding dasarnya. Karena penafsirannya berupa perbandingan, ukuran ini tidak terpengaruh oleh satuan harga sehingga dapat dibandingkan antarsaham. Ukuran inilah yang dipakai sebagai penentu utama apakah model benar-benar bermanfaat. Angka hasil perhitungannya disajikan pada Bab IV. Rumus MASE dinyatakan pada Persamaan (2.13).

$$\text{MASE} = \frac{\text{MAE}_{\text{model}}}{\text{MAE}_{\text{naive}}} \quad \text{.....(2.13)}$$

Dimana:

y = nilai sebenarnya,

y topi = nilai ramalan, dan

penyebut = rata-rata kesalahan peramalan dasar pada data latih.

Rumus ini membagi MAE model dengan MAE peramalan dasar pada data latih. Nilai di bawah satu berarti model lebih baik dari peramalan dasar, nilai tepat satu berarti setara, dan nilai di atas satu berarti lebih buruk. Karena menggunakan perbandingan, MASE tidak terpengaruh oleh skala atau satuan data sehingga cocok untuk membandingkan kinerja model antardataset yang berbeda.

2.9.5 Hipotesis Pasar Efisien Bentuk Lemah

Hipotesis pasar efisien bentuk lemah menyatakan bahwa harga yang berlaku saat ini sudah memuat seluruh informasi yang terkandung dalam harga-harga masa lalu [21]. Apabila anggapan tersebut benar, maka mempelajari pola harga masa lalu tidak akan banyak menolong untuk memperkirakan harga masa depan. Setiap pola yang menguntungkan akan segera diketahui banyak pelaku pasar, lalu dimanfaatkan sampai keuntungannya habis dengan sendirinya. Akibatnya perubahan harga dari hari ke hari menjadi sangat mendekati acak dan sulit ditebak arahnya. Anggapan ini menjelaskan mengapa ketepatan arah pada banyak penelitian peramalan saham berada di sekitar lima puluh persen. Anggapan ini juga menjelaskan mengapa model yang rumit sekalipun sering kesulitan mengalahkan peramalan dasar. Pemahaman tersebut menjadi dasar untuk menafsirkan hasil penelitian ini secara terbuka pada Bab IV.

2.9.6 Uji Binomial terhadap Peluang Lemparan Koin

Subbab 2.9.3 telah menetapkan lima puluh persen sebagai batas pembanding ketepatan arah, tetapi angka ketepatan arah yang diperoleh dari sampel terbatas selalu menyimpang sedikit dari nilai sebenarnya karena keacakan penarikan sampel. Karena itu diperlukan uji yang memastikan apakah simpangan tersebut nyata atau sekadar kebetulan. Uji binomial menguji hipotesis nol bahwa peluang memperkirakan arah dengan benar sama dengan setengah. Peluang memperoleh x perkiraan benar dari n percobaan dinyatakan pada Persamaan (2.12a).

$$\mathrm{P}(X = x) = \mathrm{C}(n, x) \times p^{x} \times (1 - p)^{n-x} \quad \text{.....(2.12a)}$$

Dimana:

n = jumlah percobaan,

x = jumlah perkiraan benar,

p = peluang benar menurut hipotesis nol yaitu 0,5, dan

C(n, x) = banyaknya kombinasi.

Proporsi perkiraan benar beserta selang kepercayaannya dinyatakan pada Persamaan (2.12b).

$$\hat{p} = \frac{x}{n} \quad \text{.....(2.12b)}$$

Dimana:

p topi = proporsi perkiraan benar yang teramati.

Nilai p di atas 0,05 berarti hipotesis nol tidak dapat ditolak, sehingga ketepatan arah model belum terbukti berbeda dari lemparan koin. Nilai p di bawah 0,05 berarti berbeda nyata, tetapi arah selisihnya wajib dinyatakan, sebab ketepatan arah yang berbeda nyata namun berada di bawah lima puluh persen justru menandakan model lebih buruk daripada perkiraan acak. Penegasan arah ini penting agar kalimat "berbeda nyata" tidak terbaca seolah-olah hasil yang baik.

2.9.7 Ukuran Pengaruh Cohen dan Uji Beda Berpasangan

Overfitting ditandai oleh kesenjangan kinerja antara data latih dan data uji, dan besar kesenjangan itu perlu diukur dengan ukuran yang tidak bergantung pada satuan. Ukuran pengaruh Cohen untuk data berpasangan dinyatakan pada Persamaan (2.13a).

$$d = \frac{\mathrm{d}}{s_d} \quad \text{.....(2.13a)}$$

Dimana:

d dengan garis atas = rata-rata selisih kinerja uji dikurangi kinerja latih pada seluruh lipatan, dan

s dengan indeks d = simpangan baku selisih tersebut.

Simpangan baku selisih dinyatakan pada Persamaan (2.13b).

$$s_d = \sqrt{\frac{\sum_k (d_k - \mathrm{d})^2}{(K-1)}} \quad \text{.....(2.13b)}$$

Dimana:

d dengan indeks k = selisih pada lipatan ke-k, dan

K = jumlah lipatan.

Statistik uji-t berpasangan dinyatakan pada Persamaan (2.13c).

$$t = \frac{\mathrm{d}}{\frac{s_d}{\sqrt{K}}} \quad \text{.....(2.13c)}$$

Nilai mutlak ukuran pengaruh Cohen di bawah 0,5 tergolong kecil, antara 0,5 dan 0,8 tergolong sedang, serta di atas 0,8 tergolong besar. Uji Shapiro-Wilk memeriksa kenormalan sebaran selisih sebagai prasyarat uji-t, dan uji Wilcoxon dipakai sebagai

pembanding yang tidak mensyaratkan kenormalan. Dengan jumlah pengamatan hanya sepuluh *fold*, daya uji ketiga alat uji tersebut sangat rendah sehingga nilai p yang besar tidak boleh dibaca sebagai bukti tidak adanya kesenjangan, melainkan hanya sebagai ketidakmampuan uji mendeteksinya. Karena itu keputusan akhir bertumpu pada ukuran pengaruh Cohen.

2.10 Penelitian Terdahulu

Penelitian tentang prediksi harga saham menggunakan pembelajaran mendalam telah berkembang pesat dalam satu dekade terakhir, dengan CNN, LSTM, dan GRU sebagai arsitektur yang paling banyak diteliti. Tinjauan terhadap berbagai karya yang telah dipublikasikan menunjukkan keragaman yang besar dalam pilihan objek, metode pra-pemrosesan, kerangka validasi, serta cara pelaporan hasil, sehingga perbandingan langsung antara satu studi dengan studi lainnya menjadi sulit dilakukan. Ringkasan dari penelitian-penelitian yang paling relevan dengan topik ini disajikan pada Tabel 2.1 berikut, yang memuat informasi mengenai peneliti dan tahun, metode yang digunakan, objek atau data, hasil utama, dan kaitannya dengan penelitian ini. Setiap entri tabel dipilih karena menggunakan setidaknya satu dari tiga arsitektur yang dibandingkan dalam penelitian ini, yaitu CNN, LSTM, atau GRU, atau karena memberikan sudut pandang metodologis yang relevan. Penelitian yang tercantum mencakup rentang waktu dari 2017 hingga 2026, mencerminkan perkembangan yang terjadi mulai dari penggunaan arsitektur tunggal hingga model hibrida yang menggabungkan beberapa arsitektur sekaligus. Perhatian khusus diberikan pada kerangka evaluasi yang digunakan oleh masing-masing penelitian, karena kerangka tersebut menentukan sejauh mana hasil yang dilaporkan dapat diandalkan dan dibandingkan secara adil.

Tabel 2.1 Ringkasan Penelitian Terdahulu tentang Prediksi Harga Saham dengan Deep Learning

No	Peneliti dan Tahun	Metode yang Digunakan	Objek/Data	Hasil Utama	Kaitan dengan Penelitian Ini

1	Selvin dkk. [9] (2017)	LSTM, RNN, CNN-sliding window	Saham India (NSE)	CNN-sliding window unggul dengan RMSE terendah. LSTM lebih baik dari RNN	Membandingkan arsitektur tunggal CNN dan LSTM. Validasi tidak menjaga urutan waktu
2	Rezaei dkk. [8] (2024)	Dekomposisi frekuensi + deep learning	Berbagai saham internasional	Hybrid dekomposisi-DL mengurangi kesalahan prediksi hingga 30% dibanding DL tunggal	Menunjukkan potensi CNN/LSTM. Tidak membandingkan arsitektur tunggal secara setara
3	Setiawan & Tjahjadi [10] (2024)	Berbagai arsitektur deep learning (CNN, LSTM, GRU)	Sektor teknologi Asia Tenggara	Hasil bervariasi antarstok. Tidak ada satu arsitektur yang selalu unggul	Perbandingan multi-arsitektur. Hyperparameter berbeda sehingga perbandingan kurang adil

4	Joshi dkk. [22] (2025)	LSTM-CNN hybrid	Saham pasar Amerika	Model hibrida mengungguli LSTM dan CNN tunggal	Menggabungkan CNN dan LSTM. Sumbangan masing-masing arsitektur sulit dipisahkan
5	Bhanujyothi & Jacob [11] (2025)	CNN-LSTM + attention + indikator teknikal	Saham pilihan (multimarket)	Akurasi prediksi meningkat dengan mekanisme perhatian	Menggunakan CNN dan LSTM. Tidak ada perbandingan arsitektur tunggal yang terkontrol
6	Chohan dkk. [12] (2026)	CNN-LSTM-GRU hybrid	Data risiko keuangan	Model hybrid memberikan deteksi risiko dini yang lebih baik	Ketiga arsitektur digabungkan. Penelitian ini menguji ketiganya secara terpisah dan setara

Berdasarkan seluruh penelitian yang telah dirangkum pada Tabel 2.1, terdapat dua kesenjangan yang belum dijawab secara memadai. Pertama, perbandingan antararsitektur tunggal jarang dilakukan pada objek yang sama dengan pengaturan pelatihan yang benar-benar disamakan, sehingga selisih kinerja yang dilaporkan dapat berasal dari perbedaan pengaturan dan bukan dari perbedaan kemampuan arsitektur itu sendiri [7]. Kedua, sebagian penelitian belum menerapkan kerangka validasi yang menjaga urutan waktu secara ketat, sehingga hasilnya berisiko terlalu optimistis akibat kebocoran data [14]. Penelitian ini berusaha mengisi kedua kesenjangan tersebut dengan menguji CNN, LSTM, dan GRU pada satu saham yang sama, yaitu GOOGL, dengan pengaturan pelatihan yang disamakan, dan

dinilai memakai 10-Fold TimeSeriesSplit yang bebas kebocoran data. Selain itu, hasil ketiga model dibandingkan dengan peramalan dasar sebagai pembanding dasar, sebuah praktik yang jarang dilaporkan pada penelitian terdahulu namun sangat penting untuk menilai apakah kerumitan model benar-benar memberikan manfaat nyata. Dengan demikian, penelitian ini menawarkan landasan evaluasi yang lebih adil dan transparan dibandingkan karya-karya sebelumnya yang diperbandingkan.