

**RANCANG BANGUN APLIKASI PREDIKSI HARGA SAHAM
MENGGUNAKAN *DEEP LEARNING* DENGAN
ALGORITMA *LONG SHORT-TERM MEMORY***

S K R I P S I

**Diajukan Sebagai Salah Satu Syarat
Untuk Kelulusan Program Sarjana S1
Program Studi Sistem Informasi**



Oleh :

Nama : Gesang Paudra Jaya

N P M : 43A87007180256

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN
KOMPUTER BANI SALEH**

BEKASI

2022

**RANCANG BANGUN APLIKASI PREDIKSI HARGA SAHAM
MENGGUNAKAN *DEEP LEARNING* DENGAN
ALGORITMA *LONG SHORT-TERM MEMORY***

S K R I P S I

**Diajukan Sebagai Salah Satu Syarat
Untuk Kelulusan Program Sarjana S1
Program Studi Sistem Informasi**



Oleh :

Nama : Gesang Paudra Jaya

N P M : 43A87007180256

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN
KOMPUTER BANI SALEH**

BEKASI

2022

LEMBAR PERSETUJUAN SKRIPSI

Telah diperiksa dan disetujui oleh pembimbing untuk disidangkan pada Sidang Skripsi Program Sarjana (S-1), Program Studi Sistem Informasi Sekolah Tinggi Manajemen Informatika dan Komputer Bani Saleh skripsi dengan judul:

RANCANG BANGUN APLIKASI PREDIKSI HARGA SAHAM MENGGUNAKAN *DEEP LEARNING* DENGAN *ALGORITMA LONG SHORT-TERM MEMORY*

Bekasi, 27 Agustus 2022

Pembimbing Utama

(Riyan Apriyanto, S.Kom, M.Kom)

Pembimbing Pendamping

(Ika Intan Rahmawati, S.Kom, M.MSI)

Mengetahui:

Ketua Program Studi Sistem Informasi



(Rudi Budi Agung, S.Si, M.MSI)

LEMBAR PENGESAHAN SKRIPSI

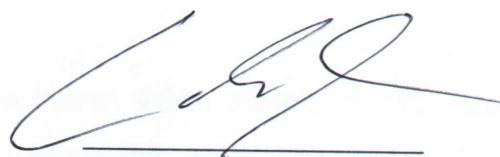
Telah disidangkan dan dinyatakan Lulus Sidang Skripsi pada Program Sarjana (S1),
Program Studi Sistem Informasi Sekolah Tinggi Manajemen Informatika dan
Komputer Bani Saleh pada tanggal bulan tahun skripsi dengan judul:

RANCANG BANGUN APLIKASI PREDIKSI HARGA SAHAM MENGGUNAKAN DEEP LEARNING DENGAN ALGORITMA LONG SHORT-TERM MEMORY

Nama Pengaji

Tanda Tangan

NAMA Adhitya Ilham Ramdhani.,
S.Kom., M.Kom.



NIDN 0406049102

NAMA Kikim Mukiman., S.Kom.,
M.Kom.



NIDN 0420117502

NAMA Budi, S.Kom., M.Kom.

NIDN 0413107601

Mengetahui:

Ketua Program Studi Sistem Informasi



(Rudi Budi Agung, S.Si, M.MSI)

PERNYATAAN KEASLIAN SKRIPSI

Nama : Gesang Paudra Jaya
NPM : 43A87007180256
Program Studi : Sistem Informasi
Judul Skripsi : Rancang Bangun Aplikasi Prediksi Harga Saham
Menggunakan *Deep Learning* Dengan Algoritma *Long Short-Term Memory*

Dengan ini saya menyatakan bahwa dalam Skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dirujuk dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila dikemudian hari saya terbukti memberikan pernyataan yang tidak benar, saya bersedia menerima sanksi berupa pencabutan gelar kesarjanaan saya.

Bekasi, 17 September 2022



(Gesang Paudra Jaya)

Gesang Paudra Jaya. 43A87007180256.

Rancang Bangun Aplikasi Prediksi Harga Saham Menggunakan Deep Learning Dengan Algroitma *Long Short-Term Memory*; dibimbing oleh Riyan Apriyanto., S.Kom., M.Kom., dan Ika Intan Rahmawati., S.Kom., M.MSI.

71 + xiii / 9 tabel/ 39 gambar/ 24 pustaka (2022)

ABSTRAK

Saham adalah suatu instrumen keuangan yang sulit untuk diprediksi, sebelumnya cara prediksi harga saham dilakukan dengan menganalisis data-data perdagangan sebelumnya secara manual, namun ini hanya berguna untuk menentukan apakah saham akan bergerak naik atau turun dengan tingkat akurasi yang tidak bisa diukur secara pasti. Oleh karena itu dibangunlah sebuah aplikasi yang menggunakan *deep learning* dengan algoritma *long short-term memory* (LSTM). Aplikasi ini menggunakan data perdagangan saham berupa harga *open*, *close*, *high*, *low*, volume perdagangan, dan kode saham agar dapat memprediksi harga *closing* kedepannya. Data yang digunakan pada aplikasi ini menggunakan data dari saham perusahaan Microsoft dan saham perusahaan Bank Central Asia (BCA), aplikasi telah diujicoba dan berhasil memprediksi harga saham dengan tingkat rata-rata akurasi keseluruhan mencapai 97,4% yang didapatkan dengan menghitung nilai rata-rata dari seluruh hasil akurasi prediksi.

Kata Kunci : *Deep Learning*, Saham, Prediksi harga saham, *Long Short-Term Memory*.

ABSTRACT

Stocks are a financial instrument that is difficult to predict. Previously, stock price prediction was done by manually analyzing previous trading data, but this is only useful for determining whether stocks will move up or down with an accuracy that cannot be measured with certainty. Therefore, an application was built that uses deep learning with a long short-term memory (LSTM) algorithm. This application uses stock trading data in the form of open, close, high, low, trading volume, and stock codes in order to predict the closing price in the future. The data used in this application uses data from shares of Microsoft companies and shares of Bank Central Asia (BCA), the application has been tested and successfully predicts stock prices with an overall average accuracy rate of 97.4% which is obtained by calculating the average value of all prediction accuracy results.

Keywords : Deep Learning, Stocks, Stocks price prediction, Long Short-Term Memory.

KATA PENGANTAR

Assalamualaikum. Wr.Wb. Puji serta syukur saya panjatkan kehadirat Allah S.W.T. Dengan rahmat, karunia dan rezekinya peneliti masih dapat hidup di dunia ini serta dengan doa dan bantuan dari kedua orang tua peneliti, peneliti dapat menyelesaikan tugas akhir berupa Skripsi yang berjudul **“Rancang Bangun Aplikasi Prediksi Harga Saham Menggunakan Deep Learning Dengan Algoritma Long Short-Term Memory”** secara tepat waktu.

Tujuan dari pembuatan Skripsi ini yang pertama adalah untuk memenuhi tugas dan tanggung jawab saya sebagai seorang mahasiswa, yang kedua adalah membuat sebuah aplikasi yang handal, terpercaya, dan sangat berguna bagi banyak pihak, peneliti berharap program atau aplikasi ini dapat terwujud dan dapat di aplikasikan kepada para pihak yang membutuhkannya.

Peneliti sungguh sangat menyadari, bahwa penelitian Skripsi ini tidak akan terwujud tanpa adanya dukungan dan bantuan dari berbagai pihak. Maka, dalam kesempatan ini peneliti menghaturkan penghargaan dan ucapan terima kasih yang sebesar-besarnya kepada :

1. Bapak Kikim Mukiman, S.Kom, M.Kom. Ketua STMIK Bani Saleh, yang telah memberikan kesempatan belajar bagi peneliti untuk dapat menyelesaikan program Sarjana di kampus tercinta ini.
2. Bapak Rudi Budi Agung, S.Si, M.MSI. Sebagai Ka Prodi Sistem Informasi STMIK Bani Saleh Bekasi, yang banyak membantu peneliti dalam mengarahkan penelitian skripsi.
3. Bapak Riyan Apriyanto, S.Kom, M.Kom, M.MSI, sebagai Pembimbing Utama Skripsi dan dosen yang dengan sabar dan tekun memberikan arahan perbaikan yang berarti bagi peneliti.
4. Ibu Ika Intan Rahmawati, S.Kom, M.MSI, sebagai Pembimbing Pendamping Skripsi dan dosen yang telah membagi ilmu pengetahuan dan pengalaman serta membimbing materi skripsi ini.

5. Rekan-rekan yang selalu membantu peneliti.

Akhir kata, dengan keterbatasan yang ada pada peneliti tentunya masih banyak kekurangan dan masih jauh dari kesempurnaan, hanya Allah SWT yang memiliki segala kesempurnaan. Oleh sebab itu masukan berupa kritik dan saran yang membangun akan sangat membantu bagi peneliti. Semoga skripsi ini dapat memberikan manfaat bagi khasanah pengetahuan Teknologi Informasi di Indonesia

DAFTAR ISI

LEMBAR PERSETUJUAN SKRIPSI.....	i
LEMBAR PENGESAHAN SKRIPSI.....	ii
PERNYATAAN KEASLIAN SKRIPSI	iii
ABSTRAK.....	iv
ABSTRACT.....	v
KATA PENGANTAR.....	vi
DAFTAR ISI	viii
DAFTAR GAMBAR.....	xi
DAFTAR TABEL	xiii
BAB I PENDAHULUAN.....	14
1.1. Latar Belakang	14
1.2. Identifikasi Masalah Dan Pembatasan Masalah.....	16
1.2.1. Identifikasi Masalah.....	16
1.2.2. Batasan Masalah.....	17
1.2.3. Rumusan Masalah	17
1.3. Tujuan Dan Manfaat Penelitian	17
1.3.1. Tujuan Penelitian.....	17
1.3.2. Manfaat Penelitian.....	18
1.4. Sistematika Penelitian	18
BAB II LANDASAN TEORI	7
2.1. Tinjauan Pustaka	7
2.2. Landasan Teori.....	8
2.2.1. <i>Artificial Neural Network</i>	8
2.2.1.1. Fungsi Aktivasi.....	11
2.2.1.2. Fungsi Loss.....	13
2.2.2. <i>Deep Learning</i>	14
2.2.3. <i>Recurrent Neural Network (RNN)</i>	15
2.2.4. <i>Long Short-Term Memory (LSTM)</i>	17
2.2.6. Google Colab.....	23
2.2.7. Google Drive	24

2.2.8.	Anvil Works	24
2.2.8.	Tensorflow	25
2.2.9.	Activity Diagram.....	25
2.2.10.	<i>Use-case</i> Diagram	27
2.4.	Kerangka Pemikiran.....	29
BAB III METODOLOGI PENELITIAN.....		30
1.1.	Analisa Kebutuhan	30
1.1.1.	Metode Pengumpulan Data	30
1.1.2.	Analisa Kebutuhan Perangkat Lunak.....	31
1.1.3.	Analisa Kebutuhan Perangkat Keras	31
1.1.3.1.	Spesifikasi Minimum Perangkat Keras.....	31
1.1.3.2.	Spesifikasi Rekomendasi Perangkat Keras	32
1.2.	Perancangan Penelitian.....	32
1.2.1.	Desain Program.....	32
1.2.1.1.	Input Data.....	36
1.2.1.2.	Data Preprocessing	37
1.2.1.3.	Membuat Model LSTM.....	37
1.2.1.4.	Training Model LSTM	38
1.2.1.5.	Output Dan Evaluasi Hasil Prediksi.....	38
1.2.2.	<i>Activity Diagram</i>	39
1.2.2.1.	Proses Latihan Model Prediksi	39
1.2.2.2.	Melihat Hasil Prediksi	41
1.2.3.	Pemodelan <i>User Interface</i>	42
1.2.4.	<i>Use-case</i> Diagram	43
1.3.	Teknik Analis	43
BAB IV HASIL DAN PEMBAHASAN		51
4.1.	Hasil	51
4.1.1.	Implementasi LSTM.....	51
4.1.1.1.	Tahap Training	51
4.1.1.2.	Tahap Evaluasi Dan Testing	53
4.1.2.	Menampilkan Hasil Prediksi Pada <i>Client-side</i>	54
4.2.	Pembahasan.....	56
4.2.1.	Hasil Proses <i>Training</i>	56

4.2.2.	Hasil Prediksi	60
4.2.3.	Hasil Pada <i>Client-Side</i>	64
4.3.	Implikasi Penelitian.....	65
BAB V KESIMPULAN DAN SARAN.....		67
1.1.	Kesimpulan	67
1.2.	Saran	68
DAFTAR PUSTAKA.....		69
DAFTAR RIWAYAT HIDUP.....		72

DAFTAR GAMBAR

Gambar	Hal
Gambar 1.1 Grafik Pergerakan Harga Saham BBCA.....	16
Gambar 1.2 Grafik Pergerakan Harga Saham MSFT	16
Gambar 2.1 Struktur <i>neuron</i> pada otak manusia	9
Gambar 2.2 Visualisasi <i>Artificial Neural Network</i>	10
Gambar 2.3 Struktur Dasar ANN	10
Gambar 2.4 Grafik Aktivasi Sigmoid.....	12
Gambar 2.5 Grafik Aktivasi TanH	13
Gambar 2.6 Struktur jaringan RNN.....	16
Gambar 2.7 Diagram LSTM	18
Gambar 2.8 <i>Forget Gate</i>	18
Gambar 2.9 <i>Input Gate</i>	19
Gambar 2.10 <i>Cell State</i>	20
Gambar 2.11 <i>Output Gate</i>	21
Gambar 2.12 Contoh Interface pada Google Colab.....	23
Gambar 2.13 Simbol Keadaan Awal	26
Gambar 2.14 Simbol Aktivitas.....	26
Gambar 2.15 Simbol <i>Action Flow</i>	26
Gambar 2.16 Simbol <i>Decision</i>	27
Gambar 2.17 Kerangka Pemikiran Aplikasi Prediksi Harga Saham.....	29
Gambar 3.1 Desain Program Secara Sederhana	33
Gambar 3.2 <i>Activity Diagram</i> Melatih Model Prediksi	40
Gambar 3.3 <i>Activity Diagram</i> Menampilkan Hasil Prediksi	41
Gambar 3.4 Desain <i>User Interface</i> Untuk <i>Client-Side</i>	42
Gambar 3.5 <i>Use-case Diagram</i> Aplikasi Prediksi Harga Saham	43
Gambar 3.6 Contoh <i>Candlestick Chart</i>	47
Gambar 3.7 Komponen <i>Candlestick</i>	47
Gambar 3.8 Pola <i>Doji</i> Pada Saham INCO.....	48
Gambar 3.9 Pola <i>Hammer</i> Pada Saham INDF	49

Gambar 3.10 Pola <i>Marubozu</i> Pada Saham ITMG	49
Gambar 4.1 Skema latihan prediksi harga saham.....	51
Gambar 4.2 Visualisasi <i>layers</i> yang digunakan pada model LSTM	52
Gambar 4.3 Contoh pembuatan UI <i>client-side</i> pada Anvil Works	55
Gambar 4.4 Grafik nilai <i>loss</i> pada hasil <i>training</i> dan validasi dengan 1000 <i>epochs</i>	57
Gambar 4.5 Grafik nilai MAE pada proses <i>training</i> dan validasi dengan 1000 <i>epochs</i>	58
Gambar 4.6 Grafik <i>plot</i> perbandingan hasil prediksi dan harga sebenarnya pada saham BBCA	59
Gambar 4.7 Grafik <i>plot</i> perbandingan hasil prediksi dan harga sebenarnya pada saham MSFT	59
Gambar 4.8 <i>Screenshot</i> dari <i>client-side</i> UI aplikasi prediksi harga saham	64
Gambar 4.9 <i>Screenshot</i> hasil prediksi saham BBCA pada <i>client-side</i>	64
Gambar 4.10 <i>Screenshot</i> hasil prediksi saham MSFT pada <i>client-side</i>	65

DAFTAR TABEL

TABEL	Hal
TABEL 2.1 Simbol-simbol pada Use-Case Diagram	28
TABEL 3.1 Parameter Yang Digunakan.....	34
TABEL 3.2 Nilai Input LSTM.....	44
TABEL 4.1 Parameter Latihan Model LSTM.....	57
TABEL 4.2 Parameter Prediksi Model LSTM.....	61
TABEL 4.3 Hasil prediksi harga saham MSFT secara harian	62
TABEL 4.4 Hasil prediksi harga saham BBCA secara harian.....	62
TABEL 4.5 Hasil Prediksi Harga Saham BBCA Dalam Rentang Waktu Berbeda	62
TABEL 4.6 Hasil Prediksi Harga Saham MSFT Dalam Rentang Waktu Berbeda.....	63

BAB I

PENDAHULUAN

1.1. Latar Belakang

Pada jaman *modern* ini kita sudah menyaksikan bahwa pada perusahaan besar maupun menengah, mereka semua telah menggunakan IT (Informasi Teknologi) berupa suatu program atau aplikasi yang membantu mereka mencapai tujuan bisnis mereka, hampir semua *stakeholder* pada suatu perusahaan menggunakan sebuah aplikasi yang membantu mereka menentukan sebuah keputusan bisnis, dengan menggunakan program atau aplikasi, maka keputusan bisnis yang dibuat menjadi lebih baik, ini menjadikan nilai perusahaan menjadi naik setiap tahun, dan saat ini hampir seluruh perusahaan besar membuka saham mereka kepada publik, dengan membuka saham mereka kepada publik maka publik dapat ikut membantu perusahaan mendapatkan lebih banyak modal untuk melakukan bisnis mereka.

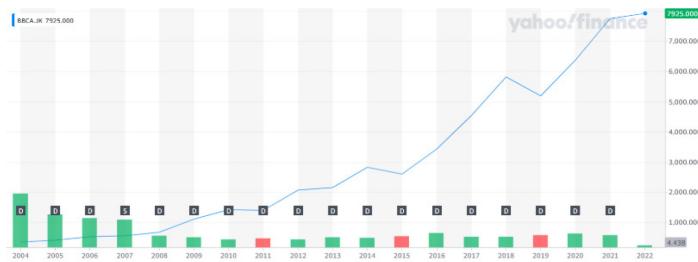
Saham adalah tanda penyertaan modal seseorang atau pihak (badan usaha) dalam suatu perusahaan atau perseroan terbatas. Dengan menyertakan modal tersebut, maka pihak tersebut memiliki klaim atas pendapatan perusahaan, klaim atas asset perusahaan, dan berhak hadir dalam Rapat Umum Pemegang Saham (RUPS) (Bursa Efek Indonesia (BEI), 2010). Naik dan turunnya nilai atau harga saham tergantung dari performa keuangan perusahaan, selain itu, keadaan sosial dan ekonomi juga bisa menjadi sentimen atau pengaruh terhadap pergerakan harga saham, maka dari itu harus dilakukan sebuah prediksi untuk menentukan apakah dalam beberapa waktu kedepan saham akan naik atau turun. Prediksi saham sendiri adalah suatu proses yang dilakukan secara sistematis untuk menentukan nilai-nilai yang paling mungkin terjadi pada saham di waktu yang akan datang berdasarkan informasi masa lalu dan informasi yang dimiliki saat ini.

Pada awalnya untuk memprediksi harga saham, para investor melihat dan mencatat data historis saham dan menentukan pola-pola pada data tersebut secara manual, namun cara ini sangat tidak praktis dikarenakan para investor secara manual harus mencatat dan memperhitungkan segala hal untuk menentukan apakah saham akan naik atau turun dalam beberapa waktu kedepan, dan itu hanya 1 saham, banyak investor yang mereka berinvestasi pada banyak saham, yang membuat tingkat kesulitan prediksi secara manual tersebut meningkat berkali lipat.

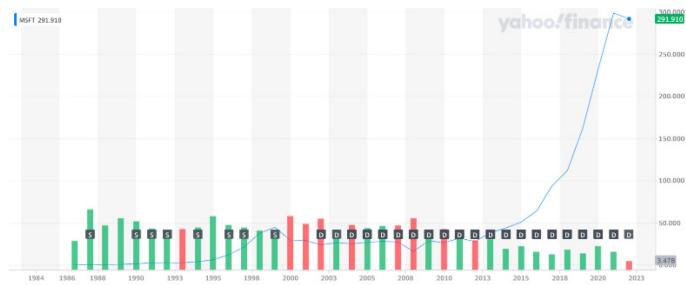
Dari permasalahan tersebut maka terbentuklah penelitian ini, yaitu sebuah program atau aplikasi untuk membantu para investor dalam mencari perusahaan yang memiliki kinerja saham dan keuangan yang baik, yaitu dimana pengguna bisa melihat apakah dalam waktu yang ditentukan harga saham perusahaan naik atau turun. Program ini peneliti buat dengan menggunakan pendekatan yang dinamakan *Deep Learning*. *Deep Learning* adalah suatu teknik pengembangan dari ML (*Machine Learning*) yang dimana sebuah program komputer membangun suatu jaringan atau *network* yang mirip dengan jaringan *neuron* pada otak manusia yang membuat proses pengembangan dan prediksi dari program lebih cepat dan lebih akurat dari sebelumnya, dan pengguna bisa menambah banyak variabel atau data untuk meningkatkan akurasi dari prediksi pengguna peneliti juga menggunakan sebuah algoritma yang ada pada *deep learning* yang dinamakan *long short-term memory* (LSTM). LSTM adalah sebuah algoritma yang memiliki *feed-back* yang mampu mengolah data sekuen sehingga bisa digunakan untuk memprediksi suatu data.

Pada penelitian ini peneliti berusaha memprediksi harga *closing* dari suatu saham, untuk dapat memprediksi harga *closing*, peneliti membutuhkan beberapa variabel dari perdagangan saham yaitu adalah *ticker* atau kode saham, *open*, *high*, *low*, *closing*, *adjclose*, volume perdagangan, semua variabel tersebut diperlukan karena semua variabel mempengaruhi satu sama lain termasuk pergerakan harga *closing* saham

sendiri, dan saham yang digunakan pada penelitian ini adalah saham MSFT dari Microsoft dan BBCA dari PT.Bank Central Asia.



Gambar 1.1
Grafik Pergerakan Harga Saham BBCA
Sumber : (Yahoo Finance, 2022)



Gambar 1.2
Grafik Pergerakan Harga Saham MSFT
Sumber : (Yahoo Finance, 2022)

1.2. Identifikasi Masalah Dan Pembatasan Masalah

1.2.1. Identifikasi Masalah

- a. Bagaimana program yang menggunakan *Deep Learning* (DL) ini dapat membantu para investor memprediksi harga saham?
- b. Bagaimana cara membuat model *Deep Learning* yang dapat digunakan untuk memprediksi harga saham?
- c. Seberapa besar tingkat akurasi model prediksi yang dibangun dengan algoritma long short-term memory (LSTM)?
- d. Bagaimana cara meningkatkan akurasi model prediksi dari program prediksi harga saham?

1.2.2. Batasan Masalah

- a. Dataset saham yang dibutuhkan diambil menggunakan API (*Application Programming Interface*) dari Yahoo Finance.
- b. Aplikasi atau program dibangun menggunakan bahasa pemrograman Python.
- c. Menggunakan modul Tensorflow 2 untuk membangun model LSTM dan mengambil *function* lain yang diperlukan.
- d. Program dijalankan dengan perangkat GPU (*Graphic Processing Unit*) agar proses latihan prediksi menjadi lebih cepat dibandingkan dengan menggunakan CPU (*Central Processing Unit*).
- e. Pembangunan program menggunakan IDE (*Integrated Development Environment*) dari Google Colab.
- f. Variabel pada saham yang digunakan adalah *ticker*, harga *close*, *open*, *adjclose*, *volume*, *high*, *low* dari saham MSFT dan BBCA.
- g. *Output* atau keluaran berupa grafik *plot* hasil prediksi dengan data sebenarnya, hasil prediksi harga closing, *loss*.

1.2.3. Rumusan Masalah

Dari penjelasan yang ada pada latar belakang, maka dapat disimpulkan suatu perumusan masalah sebagai berikut :

Bagaimana cara memprediksi harga saham dan cara membangun aplikasi prediksi harga saham menggunakan *deep learning* dengan algoritma *long short-term memory*?

1.3. Tujuan Dan Manfaat Penelitian

1.3.1. Tujuan Penelitian

- a. Mengetahui apakah saham perusahaan akan naik atau turun dalam tempo waktu yang ditentukan.
- b. Membuat program atau aplikasi yang mampu memprediksi harga saham.

- c. Mengetahui tingkat akurasi yang bisa didapatkan dari memprakirakan harga saham dengan menggunakan algoritma lstm.
- d. Mengetahui bagaimana membuat model prediksi untuk membangun program prediksi harga saham.
- e. Mengetahui bagaimana cara menggunakan algoritma lstm untuk membangun sebuah program prediksi.

1.3.2. Manfaat Penelitian

- a. Sebagai referensi bagi para investor untuk memprakirakan harga saham kedepannya.
- b. Sebagai referensi bagi para investor baru yang belum paham cara untuk menentukan saham perusahaan mana yang akan mereka beli.
- c. Sebagai referensi untuk membuat model prediksi menggunakan algoritma lstm.

1.4. Sistematika Penelitian

Untuk mempermudah melihat dan mengetahui pembahasan yang ada pada skripsi ini secara menyeluruh, maka perlu dikemukakan sistematika yang merupakan kerangka dan pedoman penelitian skripsi.

BAB I PENDAHULUAN.

Bab ini berisi penjelasan umum secara ringkas dan padat yang menggambarkan usulan isi penelitian dengan tepat yang meliputi : Latar belakang penelitian, identifikasi permasalahan yang ada pada penelitian, batasan masalah, tujuan serta manfaat dari penelitian, serta sistematika penelitian pada penelitian ini.

BAB II LANDASAN TEORI.

Bab ini berisi penjelasan mengenai landasan dari teori yang digunakan di dalam penelitian dan kerangka dari penelitian yang meliputi : Tinjauan pustaka, landasan teori, dan kerangka pemikiran.

BAB III METODOLOGI PENELITIAN.

Bab ini berisi penjelasan utama dari penelitian secara deskriptif kuantitatif, eksplanatif, eksploratif, bab ini meliputi : analisa kebutuhan-kebutuhan, rancangan penelitian, dan jadwal penelitian.

BAB IV HASIL DAN PEMBAHASAN.

Bab ini berisi hasil-hasil yang didapatkan dari penelitian dan membahasnya secara deskriptif, serta implikasi dan dampak dari penelitian, bab ini meliputi hasil penelitian, pembahasan penelitian, dan implikasi penelitian.

BAB V SARAN DAN KESIMPULAN.

Bab ini berisi rangkuman dari hasil-hasil penelitian ini, rangkuman ini berupa kesimpulan yang dijabarkan secara umum dan spesifik yang bersumber dari bab sebelumnya, yaitu hasil dan pembahasan. Bab ini juga berisi saran dari peneliti mengenai penelitian ini, saran ini berupa rekomendasi kepada para pengguna yang ingin mencoba dan menggunakan program yang dikembangkan oleh peneliti.

BAB II

LANDASAN TEORI

2.1. Tinjauan Pustaka

Penelitian mengenai bagaimana cara memprediksi saham sudah banyak dilakukan baik secara manual-matematis maupun secara program melalui komputer, salah satu penelitian prediksi harga saham dilakukan oleh beberapa dosen dari 2 perguruan tinggi di Indonesia, mereka menggunakan *Support Vector Regression* (SVR) sebagai pondasi program mereka dengan tambahan metode *Grid Search* yang mereka klaim memiliki tingkat akurasi sampai dengan 93%. Pada penelitian ini mereka hanya menggunakan fungsi *kernel linier* pada fungsi pemisah (*hyperplane*) SVR. Parameter terbaik pada fungsi kernel tersebut ditentukan dengan mencobakan beberapa nilai pada rentang tertentu untuk membangun *hyperplane*. Parameter yang dioptimalkan adalah nilai C dan nilai epsilon. Untuk menentukan parameter terbaik pada pemodelan harga saham ini digunakan metode *grid search* yang dipadukan dengan metode *leave-one-outcross-validation* untuk menghitung nilai *error* model yang dicobakan pada data training. Parameter yang terbaik untuk *hyperplane* ditentukan dengan nilai *error* terkecil. Dari seleksi parameter yang dilakukan, didapatkan parameter terbaik untuk *hyperplane* dengan fungsi *kernel linier* adalah C = 0,1 dan epsilon = 0,1 (Yasin et al., 2014).

Penelitian selanjutnya adalah penelitian prediksi harga saham dengan menggunakan metode dari turunan *Deep Learning*, yang berjudul ***Deep Learning For Stock Prediction Using Numerical And Textual Information***. Penelitian ini dilakukan oleh beberapa mahasiswa dari Universitas Kobe, Jepang. Mereka melakukan penelitian ini untuk menangani 4 hal permasalahan sekaligus, mereka menjelaskan bahwa mereka menerapkan *Vector Paragraph* untuk mendapatkan representasi terdistribusi berkelanjutan dari setiap artikel berita.

Mereka menggunakan representasi terdistribusi dan harga buku harian dari 10 perusahaan untuk memprediksi harga penutupan mereka dengan *regression analysis*. Sebagai model prediktif, mereka menggunakan model LSTM, untuk menangani pengaruh deret waktu yang ada. Eksperimen simulasi pasar saham pada kumpulan data berita keuangan menunjukkan bahwa model mereka secara efektif mengatasi masalah ini (Akita et al., 2016).

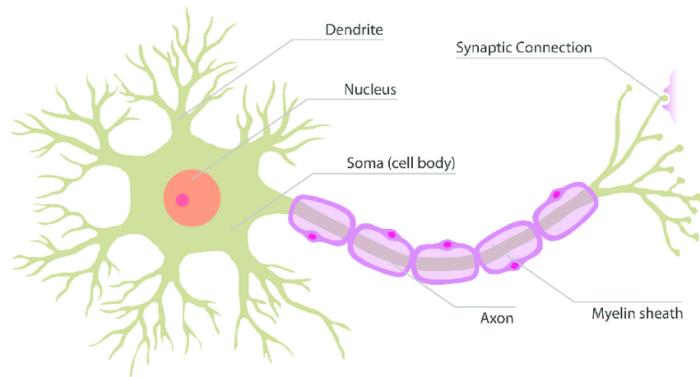
2.2. Landasan Teori

Setelah melihat beberapa penelitian yang telah dilakukan oleh peneliti tersebut, selanjutnya adalah membahas bagaimana sebuah *Artificial Intelligence* atau AI dapat membantu manusia memecahkan berbagai masalah sampai dengan permasalahan yang kita bahas saat ini, yaitu bagaimana cara memprediksi harga saham. AI memiliki banyak definisi, salah satu definisi yang terkenal adalah definisi menurut John McCarthy, yaitu “AI adalah ilmu dan rekayasa pembuatan mesin cerdas, khususnya program komputer cerdas. Ini terkait dengan tugas serupa menggunakan komputer untuk memahami kecerdasan manusia, tetapi AI tidak harus membatasi dirinya pada metode yang dapat diamati secara biologis” (McCarthy, 1956). Perkembangan lebih lanjut dari AI yang saat ini menjadi alat yang sangat berguna untuk mengolah segala data dan menyelesaikan berbagai permasalahan adalah *Deep Learning*.

2.2.1. *Artificial Neural Network*

Artificial Neural Network atau jaringan saraf buatan adalah suatu model komputasi paralel yang meniru jaringan saraf pada otak manusia, didalam otak manusia sendiri terdapat sekitar milyaran *neuron* yang saling terhubung satu sama lain, hubungan ini biasa disebut dengan *synapse*. *Neuron* terdiri dari satu inti sel yang melakukan pemrosesan informasi, satu *axon*, dan satu *dendrite*. Informasi yang masuk kedalam *neuron* akan diterima oleh *dendrite*.

Selain menerima, *dendrite* juga menggunakan *axon* sebagai keluaran dari hasil preprosesan informasi yang sudah dilakukan oleh inti sel.

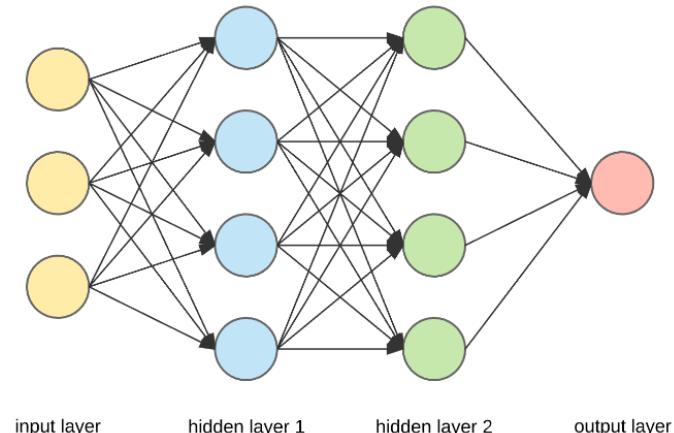


Gambar 2.1

Struktur *neuron* pada otak manusia

Sumber : (Zafeirios Fountas, 2011)

Gambar diatas adalah sebuah struktur *neuron* yang disederhanakan, cara kerja dari *neural network* atau jaringan saraf adalah dimulai pada sinyal yang masuk melalui *dendrite*, kemudian diproses oleh *cell body* dengan fungsi tersendiri. Informasi yang telah diproses ini kemudian dikirim ke *neuron* lain melalui *synaptic connection*, jika sinyal pada informasi yang telah diproses memenuhi nilai ambang batas atau *threshold*, maka informasi tersebut akan diterima. Pada kasus tersebut sebuah *neuron* bisa dikatakan sudah diaktivasi. ANN atau *Artificial Neural Network* merupakan sistem adaptif yang dapat mengubah strukturnya untuk memecahkan suatu masalah berdasarkan informasi internal dan informasi eksternal. Seperti pada otak manusia, selalu memiliki kemampuan untuk belajar dengan melakukan adaptasi (Rosebrock, 2017).

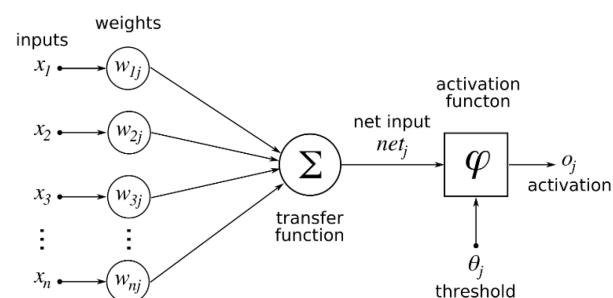


Gambar 2.2

Visualisasi *Artificial Neural Network*

Sumber : (Dhea Larasati, 2019)

Neural network memiliki beberapa tipe yang berbeda satu sama lainnya, namun hampir semua komponen utama dan karakteristik yang dimiliki indentik satu sama lain, ini sama dengan jaringan saraf pada otak manusia, *artificial neural network* terdiri dari beberapa *neuron* dan antar *neuron* juga terhubung satu sama lain, semua *neuron* tersebut akan memproses informasi dan diterima *neuron* lain melalui sambungan antar satu *neuron* dengan *neuron* yang lain berdasarkan nilai bobot atau *weight* yang dimiliki antar *neuron*.



Gambar 2.3

Struktur Dasar ANN

Sumber : (BD TechTalk, 2019)

Pemecahan suatu masalah menggunakan *artificial neural network* tidak memerlukan suatu pemrograman yang terus menerus, jaringan saraf ini menyelesaikan masalah melalui proses dari *learning* atau *training* dari contoh-contoh yang diberikan. Pada jaringan ANN biasanya diberikan sebuah pola pelatihan yang terdiri dari sekumpulan contoh pola, proses *training* pada ANN berasal dari serangkaian contoh-contoh pola yang diberikan. Metode *training* yang sering dipakai adalah metode *supervised learning*. Selama proses *training* itu pola *input* dimasukkan secara paralel atau bersama-sama dengan pola *output* yang diinginkan. ANN akan menyesuaikan nilai bobotnya sebagai tanggapan atas pola *input* dan *output* yang diberikan.

2.2.1.1. Fungsi Aktivasi

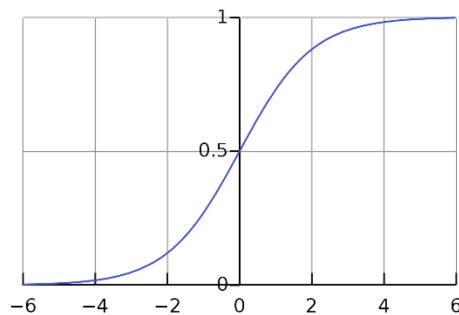
Fungsi aktivasi atau *activation function* adalah sebuah fungsi yang menggambarkan hubungan antara tingkat aktivitas *internal* yang mungkin terbentuk linear atau non-linear yang berfungsi untuk menentukan apakah *neuron* tersebut harus aktif atau tidak berdasarkan nilai *weighted sum* yaitu nilai bobot dari hasil penjumlahan yang berasal dari *input* (Mubarok, 2019). Berikut adalah beberapa fungsi aktivasi yang sering digunakan :

a. Sigmoid

Sigmoid adalah fungsi aktivasi yang memiliki nilai antara 0 dan 1, karena nilainya antara 0 dan 1, maka fungsi sigmoid ini paling cocok untuk memprediksi sebuah kemungkinan atau probabilitas. Berikut adalah karakteristik dari fungsi sigmoid :

- 1) Fungsi dapat terdiferensialkan, artinya pengembang dapat menemukan kemiringan kurva sigmoid pada dua titik manapun.
- 2) Fungsinya monoton, tetapi turunan fungsinya tidak.

Formula Sigmoid adalah : $g(x) = \frac{1}{1+e^{-x}}$ (1)



Gambar 2.4

Grafik Aktivasi Sigmoid

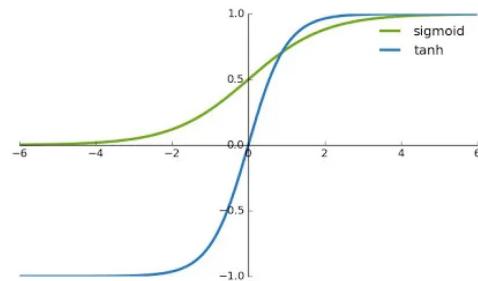
Sumber : (Sagar Sharma, 2017)

b. TanH atau Fungsi *Hyperbolic Tangent Activation*

TanH adalah fungsi yang mirip dengan sigmoid namun lebih baik, rentang nilai pada fungsi ini mulai dari -1 sampai 1. Dengan memiliki nilai negatif maka *input* yang bernilai negatif bisa akan dipetakan sangat negatif dan *input* nol akan dipetakan mendekati nol pada grafik tanh. Berikut adalah karakteristik fungsi TanH :

- 1) Fungsi dapat terdiferensialkan.
- 2) Fungsinya monoton, tetapi turunan fungsinya tidak.
- 3) Fungsi TanH biasanya digunakan untuk klasifikasi antara dua kelas atau kategori.

Formula pada fungsi TanH adalah : $g(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ (2)



Gambar 2.5

Grafik Aktivasi TanH

Sumber : (Sagar Sharma, 2017)

2.2.1.2. Fungsi *Loss*

Fungsi *loss* atau *loss function* adalah sebuah fungsi yang mengkalkulasikan jarak antara *output* saat ini dengan *output* yang diinginkan. Ini adalah metode untuk mengevaluasi bagaimana performa algoritma yang ada pada model data. *Loss* dapat dibagi dalam dua kategori, yaitu untuk klasifikasi, dan regresi.

a. *Mean Absolute Error (MAE)*

Adalah nilai mutlak untuk rata-rata selisih nilai sebenarnya dengan nilai prediksi, MAE digunakan untuk mengukur keakuratan suatu model dalam melakukan prediksi.

$$MAE = \frac{1}{n} \sum_{i=1}^n |A^i - F^i| \quad (3)$$

n = Ukuran sampel

F^i = nilai data prediksi ke-i

A^i = nilai data asli ke-i

Berdasarkan rumus MAE diatas, bahwa komponen $|A^i - F^i|$ menunjukkan nilai perbedaan pada nilai sebenarnya dengan nilai prediksi. Para pengembang ingin bahwa nilai perbedaan ini semakin kecil.

Semakin kecil nilai MAE maka semakin baik model dalam melakukan prediksi.

2.2.2. Deep Learning

Deep learning adalah bagian dari pembelajaran mesin atau *Machine Learning*, yang pada dasarnya adalah *neural network* atau jaringan saraf dengan tiga atau lebih lapisan. Jaringan saraf ini mencoba untuk mensimulasikan perilaku otak manusia meskipun jauh dari kemampuan yang memungkinkannya untuk belajar dari sejumlah besar data. Sementara jaringan saraf dengan satu lapisan masih dapat membuat perkiraan perkiraan, lapisan tersembunyi tambahan dapat membantu mengoptimalkan dan menyempurnakan akurasi (IBM, 2020a).

Deep learning biasanya membutuhkan *dataset* dalam jumlah yang besar, dengan begitu tingkat akurasi yang tinggi akan tercapai, dengan menggunakan *dataset* yang cukup besar membuat *deep learning* membutuhkan sebuah *hardware* dengan komputasi yang tinggi untuk meningkatkan kecepatan pemrosesan latihan, ini yang menyebabkan *deep learning* saat ini membutuhkan sebuah GPU atau *Graphic Processing Unit* yang saat ini merupakan sebuah *hardware* dengan komputasi paralel tercepat yang mampu membantu komputasi *deep learning* berjalan lebih cepat daripada menggunakan sebuah atau beberapa CPU (*Central Processing Unit*). *Deep learning* sendiri saat ini banyak digunakan untuk membantu manusia mulai dari sektor bisnis, industri, bahkan sampai sektor kesehatan dan farmasi.

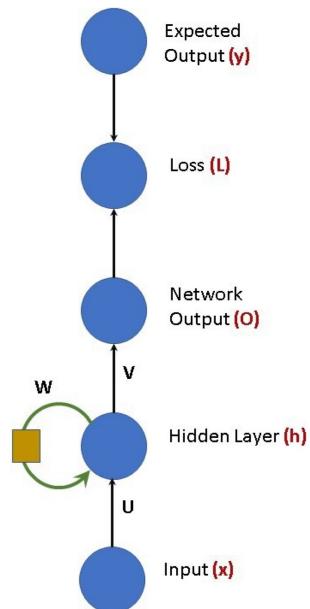
Deep learning sendiri memiliki banyak algoritma, beberapa yang sering digunakan adalah CNN atau *Convolutional Neural Network* yang dapat mengolah gambar dan video untuk analisis, dan klasifikasi, kemudian ada GANs atau *Generative Adversarial Networks* yang dapat memanipulasi gambar, mensimulasikan video dengan data gambar yang diberikan, dan kemudian ada Boltzman *Machine* yang dapat belajar dari distribusi probabilitas yang diberikan dan digunakan untuk regresi, klasifikasi, *feature learning* dan *filtering*.

Salah satu algoritma *deep learning* yang digunakan untuk memprediksi atau memprakirakan sebuah data adalah RNN atau *Recurrent Neural Network* yang mampu memprediksi suatu data, namun RNN memiliki keterbatasan yang membuat RNN tidak bisa untuk membuat prediksi yang terlalu jauh dimasa mendatang, permasalahan ini disebut dengan *vanishing gradient* atau *exploding*.

2.2.3. *Recurrent Neural Network* (RNN)

Seperti yang telah dijelaskan sebelumnya, RNN adalah sebuah algoritma dari *deep learning* yang biasanya digunakan untuk membuat suatu prakiraan atau prediksi, RNN adalah sebuah *neural network* yang menggunakan data *time series*, algoritma ini biasanya digunakan untuk permasalahan ordinal atau temporal, seperti menerjemahkan bahasa, *natural processing language* (NLP), *speech recognition*, *image captioning*. RNN juga terdapat pada aplikasi populer yaitu Cortana, Google Assistant, Siri, dan Google Terjemahan. Seperti pada CNN, RNN memanfaatkan data *training* untuk belajar, RNN dibedakan dengan jaringan lainnya karena RNN memiliki sebuah “memori” karena RNN mengambil informasi dari *input* sebelumnya untuk menentukan atau mempengaruhi *output* saat ini, sementara jaringan saraf lain mengasumsikan bahwa *input* dan *output* adalah independen satu sama lain, *output* pada RNN

tergantung dari *input* sebelumnya dalam urutan. Sementara peristiwa yang akan datang juga akan membantu dalam menentukan *output* dari urutan yang diberikan. RNN searah tidak dapat menjelaskan peristiwa ini didalam prediksinya, hanya RNN *bidirectional* yang bisa (IBM, 2020).



Gambar 2.6
Struktur jaringan RNN
Sumber : (Rastogi, 2020)

Karakteristik lain pada RNN adalah RNN dapat membagikan parameter ke semua *layer* yang ada pada jaringan. Sementara *feed-forward network* mempunyai perbedaan *weights* pada setiap *node*, RNN berbagi parameter *weights* yang sama pada masing-masing *layer* didalam jaringan. Namun, *weights* ini tetap diubah pada setiap proses *backpropagation* dan menentukan gradien untuk meningkatkan performa latihan.

RNN menggunakan algoritma *backpropagation through time* (BPTT) untuk menentukan gradien, yang berbeda dari *backpropagation* biasa, karena ini khusus untuk data *sequence* atau urutan. Prinsip BPTT ini sama dengan BPTT non urutan, dimana

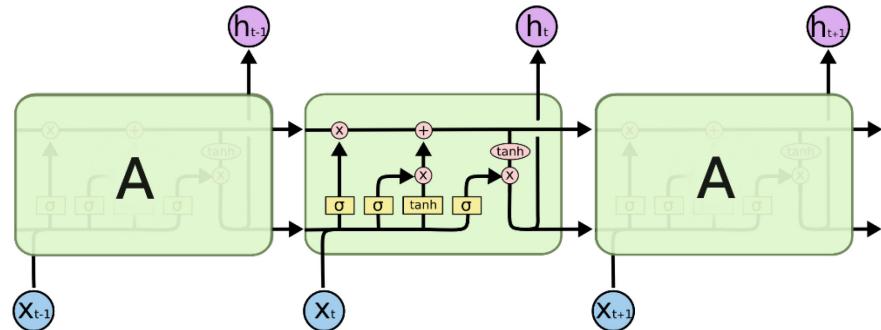
model melatih dirinya sendiri dengan menghitung kesalahan dari lapisan *output* sampai lapisan *input*. Perhitungan ini memungkinkan peneliti untuk menyesuaikan parameter model dengan tepat. Untuk BPTT pada RNN ini, algoritma ini menjumlahkan kesalahan pada setiap *time step* sedangkan *feed-forward network* tidak perlu menjumlahkan kesalahan karena tidak adanya pembagian parameter pada setiap *layer* atau lapisan.

Karena proses tersebut, RNN memiliki 2 masalah utama, yang pertama adalah *exploding gradient* dan yang kedua adalah *vanishing gradient*, Permasalahan ini ditentukan dengan ukuran gradien, yang merupakan kemiringan fungsi *loss* di sepanjang kurva *error*. Ketika gradien terlalu kecil, maka akan menjadi lebih kecil lagi, sehingga memperbarui parameter *weights* menjadi sulit bahkan tidak mungkin untuk memperbarui parameter *weights* tersebut, dikarenakan gradien ini mendekati 0 atau akan menjadi 0, ketika ini terjadi, maka RNN akan berhenti dari latihan pembelajarannya, maka terjadilah *gradient explosion* yang menciptakan model yang tidak stabil. Dalam hal ini, *weights* pada model akan tumbuh terlalu besar, dan pada akhirnya menjadi sebuah NaN (*Not a Number*). Salah satu solusi dari hal ini adalah dengan mengurangi jumlah *hidden layer* pada jaringan, dan membuang semua jaringan kompleks pada model RNN.

2.2.4. Long Short-Term Memory (LSTM)

LSTM memiliki *feedback connection* yang membuatnya berbeda dengan *recurrent neural network*. Properti ini memungkinkan LSTM untuk memproses seluruh urutan data (misalnya deret waktu) tanpa memperlakukan setiap titik dalam urutan secara independen, melainkan, mempertahankan informasi yang berguna tentang data sebelumnya dalam urutan untuk membantu pemrosesan titik data baru. Akibatnya, LSTM sangat baik

dalam memproses urutan data seperti teks, ucapan, dan deret waktu (Dolphin, 2020). LSTM juga adalah sebuah solusi dari permasalahan *vanishing gradient* yang ada pada RNN.

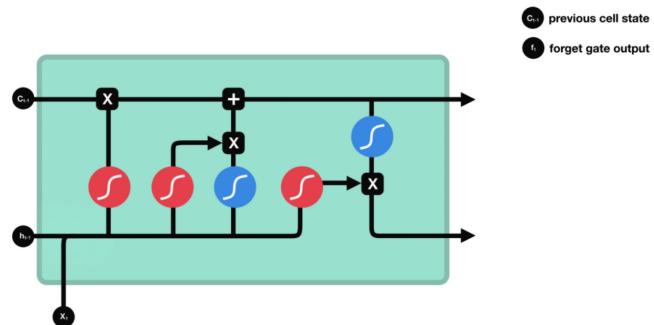


Gambar 2.7

Diagram LSTM

Sumber : (Colah, 2015)

LSTM menggunakan beberapa *gate* untuk mengontrol bagaimana informasi didalam data masuk, disimpan, dan keluar dari jaringan, ketiga *gate* tersebut adalah *input gate*, *output gate*, *forget gate*. Ketiga gerbang ini juga dianggap sebagai *filter* dan masing-masing memiliki *neural network* sendiri.



Gambar 2.8

Forget Gate

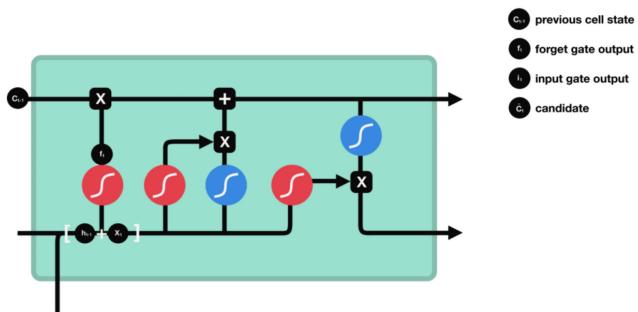
Sumber : (Phi, 2018)

a. *Forget Gate*

Forget gate atau gerbang lupa adalah *gate* yang menentukan apakah data *input* X_t dan *output* h_{t-1} bisa lewat atau tidak. Keputusan ini dibuat oleh fungsi *sigmoid* yang menghasilkan nilai f_t . Pada *output* ini, nilai mendekati 1 berarti data “boleh lanjut” atau lewat, sedangkan mendekati 0 berarti data “tidak boleh lewat” atau “abaikan”.

$$f_t = \sigma(U_f X_t + W_f h_{t-1} + b_f) \quad (4)$$

b. *Input Gate*



Gambar 2.9

Input Gate

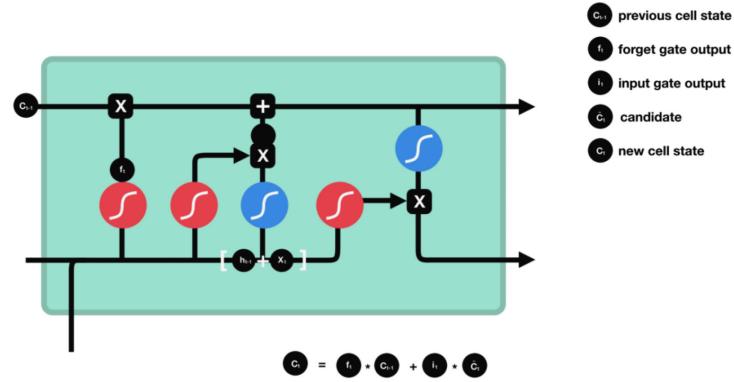
Sumber : (Phi, 2018)

Input Gate adalah *gate* yang menentukan bagian mana pada jaringan yang akan diperbarui atau *update*, *input gate* memiliki dua bagian *layer*, yaitu *layer* dengan fungsi *sigmoid* dan *layer* dengan fungsi *tanh*. *Layer* dengan fungsi *sigmoid* menentukan nilai mana yang diperbarui dan menghasilkan nilai i_t kemudian *layer* dengan fungsi *tanh* memberikan *weight* atau bobot pada nilai-nilai yang dilewati, kemudian menentukan tingkat kepentingan dan menghasilkan nilai \tilde{c}_t . Kemudian kedua fungsi ini bisa digabung dengan cara mengkalikan kedua nilai, selanjutnya adalah mengkalikan C_{t-1} kondisi sebelumnya

dengan nilai *forget gate* f_t . Setelah itu menambahkan dengan hasil perkalian i_t dan \tilde{c}_t sehingga menciptakan *cell state* C_t (Aprian, 2020)

$$i_t = \sigma(U_i X_t + W_i h_{t-1} + b_i) \quad (5)$$

c. Cell State



Gambar 2.10

Cell State

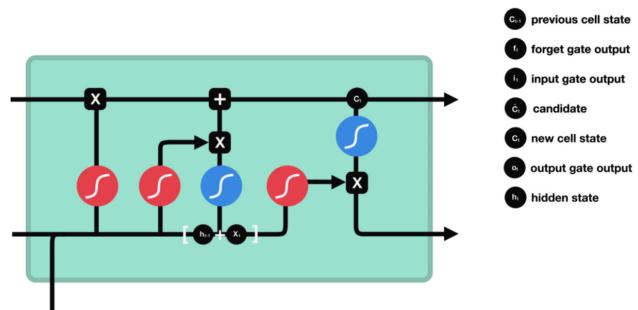
Sumber : (Phi, 2018)

Selanjutnya memperbarui nilai cell state lama c_{t-1} menjadi cell state baru \tilde{C}_t dan C_t

$$\tilde{C}_t = \tanh(U_c X_t + W_c h_{t-1} + b_c) \quad (6)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (7)$$

d. Output Gate



Gambar 2.11

Output Gate

Sumber : (Phi, 2018)

Terakhir adalah *output gate*, *gate* ini memutuskan apa yang akan dihasilkan, pertama menjalankan fungsi *sigmoid* untuk menentukan *cell* apa yang akan dihasilkan pada O_t . Selanjutnya mengkalikan dengan *cell state* C_t yang sudah melewati fungsi *tanh* dan hasilnya menjadi h_t kemudian meneruskannya ke *cell* selanjutnya yaitu h_{t-1} . *Output gate* ini tidak berkontribusi untuk *cell state*, namun *gate* inilah yang membedakan *cell state* dan *output* h_t yang sebenarnya (Aprian, 2020)

$$O_t = \sigma(U_o X_t + W_o h_{t-1} + b_o) \quad (8)$$

$$h_t = O_t * \tanh(C_t) \quad (9)$$

Keterangan :

f_t = *Forget Gate*

i_t = *Input Gate*

C_t = *Cell State*

O_t = *Output Gate*

X_t = *Input*

h_t = *Output*

h_{t-1} = *Output Sebelumnya*

C_{t-1} = *Cell State Sebelumnya*

σ = *Sigmoid*

b = *Bias*

W = *Weight*

U = *Weight*

\tanh = *Tanh*

Semua tahap tersebut digunakan berulang kali, contohnya adalah jika LSTM digunakan untuk memprediksi harga saham berdasarkan data 30 hari terakhir, maka semua proses akan diulang selama 30 kali atau $t = 30$. Namun *output* masih didalam *hidden state*, kita tidak bisa menggunakan *hidden state* sebagai *output* terakhir, maka dari itu untuk mengubah *hidden state* menjadi *output* yang diinginkan, kita menerapkan *layer* linear yang menjadi tahap paling akhir pada LSTM. *Layer* atau lapisan ini hanya diterapkan sekali, dipaling akhir dari semua lapisan pada model LSTM.

Selanjutnya untuk menentukan akurasi dari prediksi yang dihasilkan, maka peneliti menggunakan formula akurasi sebagai berikut :

$$A = 1 - \left| \frac{N_p}{N_a} \right| \times 100\% \quad (10)$$

$A = \text{Nilai akurasi dalam persen}$

$N_p = \text{Nilai Prediksi}$

$N_a = \text{Nilai Sebenarnya}$

Kemudian untuk menentukan rata-rata keseluruhan akurasi prediksi, maka peneliti menggunakan formula sebagai berikut :

$$R_t = \frac{\bar{x}}{n} \times 100\% \quad (11)$$

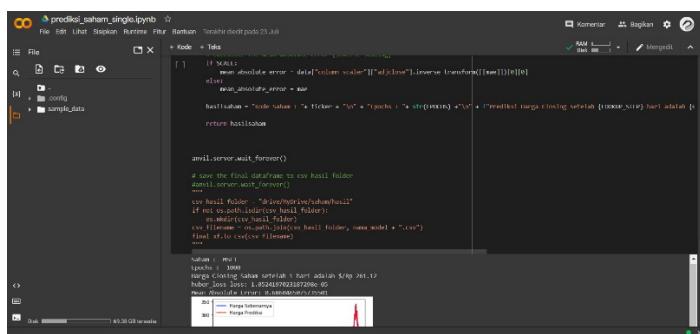
$R_t = \text{Nilai rata - rata keseluruhan akurasi prediksi}$

$\bar{x} = \text{Jumlah total akurasi prediksi}$

$n = \text{Banyaknya data akurasi prediksi}$

2.2.6. Google Colab

Google Colab adalah sebuah IDE (*Integrated Development Environment*) berupa python *notebook* yang digunakan untuk membuat, menjalankan program yang menggunakan bahasa pemrograman python, Google Colab memungkinkan siapa saja untuk menulis dan mengeksekusi kode python melalui browser, dan sangat cocok untuk *machine learning*, analisis data, dan pendidikan. Secara lebih teknis, Colab adalah layanan notebook Jupyter yang dihosting yang tidak memerlukan penyiapan untuk digunakan, sekaligus memberikan akses gratis ke sumber daya komputasi termasuk GPU. (Google, 2017).



Gambar 2.12

Contoh Interface pada Google Colab

Colab memiliki versi gratis dan berbayar, versi gratis dari Google Colab memiliki beberapa batasan, google hanya mengijinkan pengguna menggunakan Google Colab versi gratis selama 12 jam dalam 24 jam, untuk versi gratis dengan GPU, google hanya mengijinkan selama 4 jam dalam 24 jam, jika pengguna membuka Colab namun tidak ada interaksi sama sekali dalam 90 menit, maka google akan memberhentikan dan menghapus *runtime* yang pengguna gunakan pada Colab. Peneliti sendiri menggunakan Google Colab versi gratis, versi ini sudah cukup untuk menjalankan program yang peneliti kembangkan, peneliti menggunakan Colab dengan *runtime* CPU untuk menampilkan hasil prediksi harga

saham, sedangkan *runtime* GPU untuk menjalankan proses latihan model prediksi, proses latihan model prediksi memakan waktu sekitar 20-40 menit dengan menggunakan Colab versi gratis, google sendiri tidak memberikan detail mengenai GPU apa yang diberikan oleh mereka kepada pengguna Colab versi gratis. Namun beberapa pengguna aplikasi yang menggunakan Colab berpendapat bahwa kemungkinan Google menggunakan GPU NVIDIA versi dengan arsitektur MAXWELL atau seri GTX 900 kebawah.

2.2.7. Google Drive

Google Drive adalah sebuah *web service* atau layanan berbasis *website* dari Google yang digunakan untuk menyimpan data dan informasi *digital* dalam format apapun yang bisa tersinkronisasi dengan perangkat pengguna. Google Drive menyediakan pengguna 15 GB tempat penyimpanan gratis melalui Google One. Google One juga menyediakan penyimpanan sebesar 100 GB, 200 GB, 2 TB, 10 TB, 20 TB, dan 30 TB, yang disediakan secara berbayar dalam harga yang berbeda-beda.

2.2.8. Anvil Works

Anvil Works adalah sebuah *drag and drop web app builder* yang memungkinkan para pengembang aplikasi membuat aplikasi berbasis *website* dengan mudah, aplikasi yang dikembangkan dengan anvil menggunakan bahasa pemrograman python, Anvil juga memungkinkan pengembang membuat *user interface* dan *client-side* dari aplikasi yang mereka kembangkan, pengembang dapat menghubungkan *backend* aplikasi python pada server pengembang dengan *client-side* pada anvil dengan mudah, pengembang dapat menghubungkan *client-side* pada anvil dengan server dari berbagai *cloud service* seperti Microsoft Azure, Google Cloud, Amazon Web Services, IBM Cloud. Peneliti menggunakan Anvil untuk membuat

client-side dengan *user interface* yang memudahkan pengguna untuk melihat hasil prediksi harga saham.

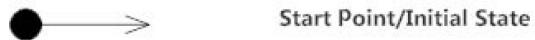
2.2.8. Tensorflow

Tensorflow adalah *open source software library* yang dikembangkan oleh Google untuk mengembangkan perangkat lunak *machine learning*, *deep learning*, dan aplikasi yang menggunakan AI lainnya, Tensorflow dapat dijalankan pada CPU, GPU, TPU(*Tensor Processing Unit*), pengembang dapat mengembangkan aplikasi dengan tensorflow pada sistem operasi Linux, Windows, MacOS, Android. Komputasi pada tensorflow dinyatakan dengan grafik aliran data *stateful*. Nama Tensorflow sendiri berasal dari sebuah operasi yang dilakukan didalam *neural network* pada tensorflow yang menggunakan array multidimensi yang disebut dengan tensor (Tensorflow & Google, 2012).

2.2.9. Activity Diagram

Activity Diagram atau diagram aktivitas adalah sebuah diagram yang secara visual menyajikan serangkaian tindakan atau aliran kontrol didalam sistem yang mirip dengan diagram *flowchart*, diagram aktivitas sering digunakan dalam pemodelan proses bisnis. Para pengguna diagram ini juga dapat menjelaskan langkah-langkah yang ada pada diagram *use case*, aktivitas yang dimodelkan dapat berurutan dan bersamaan. Dalam kedua kasus, diagram aktivitas akan memiliki sebuah keadaan awal atau *initial state* dan keadaan akhir atau *final state* (Smart Draw, 2019).

- a. Keadaan awal atau *initial state*



Gambar 2.13

Simbol Keadaan Awal

Sumber : (Smart Draw, 2019)

Lingkaran kecil yang diikuti oleh panah mewakili status tindakan awal atau titik awal untuk diagram aktivitas apa pun. Untuk diagram aktivitas menggunakan *swimlanes*, pastikan titik awal diletakkan di pojok kiri atas kolom pertama (Smart Draw, 2019).

- b. Aktivitas



Gambar 2.14

Simbol Aktivitas

Sumber : (Smart Draw, 2019)

Adalah Status tindakan mewakili tindakan objek yang tidak dapat diinterupsi (Smart Draw, 2019).

- c. *Action Flow*



Gambar 2.15

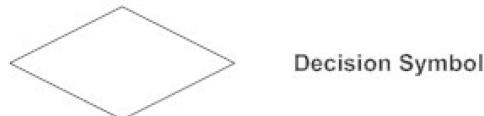
Simbol *Action Flow*

Sumber : (Smart Draw, 2019)

Action Flow atau alur aksi, bisa juga disebut *edge and path*, adalah sebuah simbol yang mengilustrasikan transisi dari satu

status aksi ke status aksi lainnya. Mereka biasanya digambar dengan garis panah (Smart Draw, 2019).

d. *Decision*



Gambar 2.16

Simbol *Decision*

Sumber : (Smart Draw, 2019)

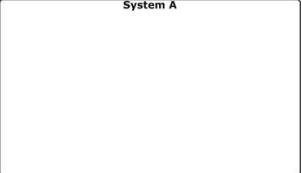
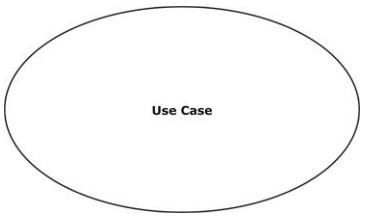
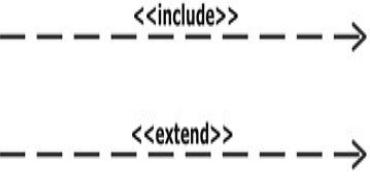
Sebuah simbol berlian yang mewakili keputusan dengan jalur alternatif. Ketika suatu aktivitas memerlukan keputusan sebelum melanjutkan ke aktivitas berikutnya, tambahkan simbol ini diantara kedua aktivitas tersebut, jalur alternatif harus diberi label dengan sebuah kondisi atau *condition*, pengguna juga bisa memberi label salah satu jalur dengan kata *else* (Smart Draw, 2019).

2.2.10. *Use-case* Diagram

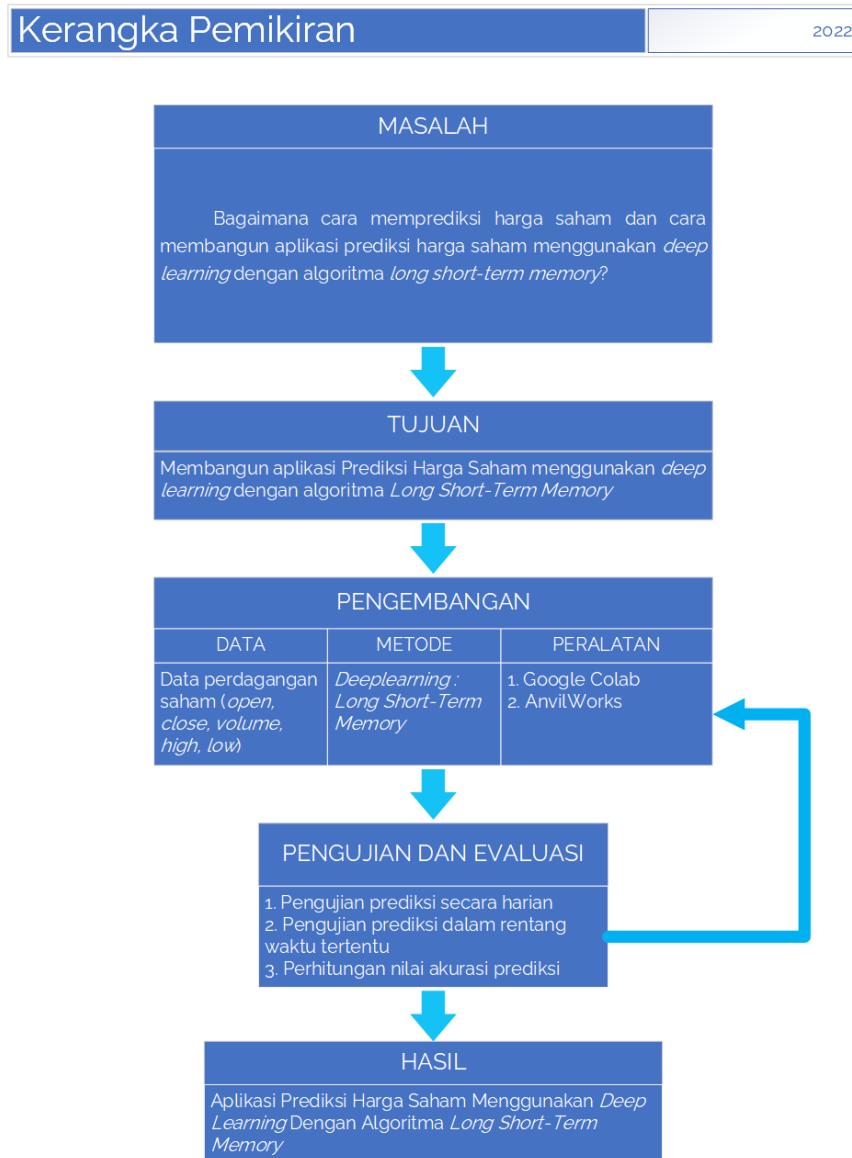
Use-case diagram adalah diagram berjenis UML (*Unified Modelling Language*) yang dapat digunakan untuk meringkas detail pengguna sistem yang disebut sebagai aktor dan interaksi mereka dengan sistem. Diagram dibuat dengan menggunakan satu set simbol dan konektor khusus (Lucid Chart, 2019). Diagram *use-case* yang efektif dapat membantu tim pengembang mendiskusikan dan memvisualisasikan :

- a. Skenario didalam sistem dimana sistem atau aplikasi berinteraksi dengan para penggunanya, dan sistem eksternal.
- b. Sasaran yang dicapai oleh sistem
- c. Cakupan Sistem

TABEL 2.1
Simbol-simbol pada *Use-Case Diagram*
Sumber : (EdrawMax, 2020)

Simbol	Fungsi
	Sistem Batas persegi panjang adalah sistem. Use case berada di dalamnya, dan aktor akan ditempatkan di luarnya.
	Use Case Bentuk oval mewakili use case. Use case mewakili fungsionalitas sistem, serta tujuan akhir dari aktor. Use case harus ditempatkan di dalam sistem.
	Aktor Ketika seorang aktor berinteraksi dengan sistem, ini memicu proses penggunaan. Aktor harus ditempatkan di luar sistem.
	Hubungan Panah digunakan untuk menunjukkan hubungan antara aktor dan use case, atau antara dua use case. <i>Extension</i> menunjukkan bahwa satu kasus penggunaan dapat mencakup perilaku kasus penggunaan lain. <i>Inclusion</i> mewakili satu kasus penggunaan menggunakan fungsionalitas kasus penggunaan lain.

2.4. Kerangka Pemikiran



©Gesang Paudra Jaya | 2022

Gambar 2.17
Kerangka Pemikiran Aplikasi Prediksi Harga Saham

BAB III

METODOLOGI PENELITIAN

1.1. Analisa Kebutuhan

Analisa kebutuhan sangat diperlukan dalam mendukung kinerja aplikasi, apakah aplikasi yang dibuat telah sesuai dengan kebutuhan atau belum. Karena kebutuhan sistem akan mendukung tercapainya tujuan suatu aplikasi.

1.1.1. Metode Pengumpulan Data

Metode pengumpulan data dalam pembangunan program ini adalah dengan melakukan pengambilan data perdagangan saham mulai dari saham tersebut dibuka untuk publik atau IPO, sampai dengan perdagangan terakhir atau terbaru, peneliti saat ini menggunakan 2 saham yaitu saham perusahaan Microsoft dengan kode saham MSFT, dan saham perusahaan BCA (Bank Central Asia) dengan kode saham BBCA atau BBCA.JK jika pengguna mengambil data saham dari luar negeri, saham milik Microsoft adalah saham yang terdaftar pada bursa saham NASDAQ yang berlokasi di New York, Amerika Serikat, kemudian saham milik BCA adalah saham yang terdaftar pada bursa saham Indeks Harga Saham Gabungan (IHSG) yang berlokasi di Jakarta, Indonesia. Data saham didapatkan menggunakan API dari Yahoo! Finance, dengan menggunakan API Yahoo! Finance maka peneliti dapat dengan mudah mendapatkan data saham dan memasukkannya kedalam program.

Untuk mengambil data tersebut, peneliti tidak perlu menggunakan akun, hanya dengan memasang API Yahoo Finance pada python *environment* maka peneliti dapat mengambil data saham dengan mudah.

1.1.2. Analisa Kebutuhan Perangkat Lunak

Jika pengguna ingin menjalankan program ini diluar Google Colab, yaitu pada komputer lain atau komputer milik pengguna, maka terdapat spesifikasi minimum perangkat lunak atau *software*, spesifikasi minimum perangkat lunak adalah perangkat lunak yang dibutuhkan agar program dapat berjalan, jika spesifikasi minimum tidak terpenuhi, maka kemungkinan besar program tidak akan berjalan sama sekali. Berikut adalah spesifikasi minimum perangkat lunak yang dibutuhkan :

- a. Sistem Operasi Ubuntu 20.04 atau Windows 10
- b. Python 3.8
- c. Anaconda 3
- d. Jupyter Lab
- e. Tensorflow 2.8
- f. Yahoo Finance API
- g. NVIDIA GPU Driver
- h. NVIDIA CUDA 11
- i. NVIDIA CUDA SDK

1.1.3. Analisa Kebutuhan Perangkat Keras

Perangkat keras atau *hardware* adalah perangkat utama yang dibutuhkan untuk menjalankan program ini, terdapat spesifikasi minimum dan spesifikasi rekomendasi pada kategori perangkat keras ini. Berikut adalah spesifikasi minimum dan spesifikasi rekomendasi perangkat keras :

1.1.3.1. Spesifikasi Minimum Perangkat Keras

Spesifikasi minimum perangkat keras adalah perangkat keras yang dibutuhkan untuk menjalankan program ini, jika perangkat keras tidak memenuhi atau kurang dari spesifikasi minimum, maka kemungkinan besar program

ini tidak akan berjalan. Berikut adalah spesifikasi minimum perangkat keras yang dibutuhkan :

- a. GPU dari NVIDIA dengan minimum VRAM sebesar 8GB dan arsitektur MAXWELL.
- b. 8GB RAM.
- c. Minimum sisa penyimpanan sebesar 32GB.

1.1.3.2. Spesifikasi Rekomendasi Perangkat Keras

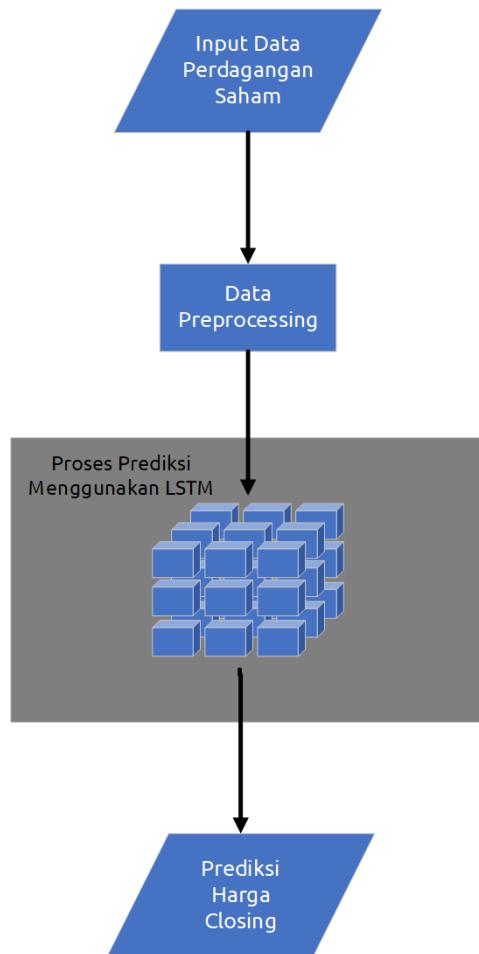
Spesifikasi rekomendasi perangkat keras adalah perangkat keras yang dibutuhkan agar program berjalan lebih cepat, pengguna masih bisa menjalankan program jika spesifikasi rekomendasi tidak terpenuhi, asalkan spesifikasi minimum sudah terpenuhi, namun jika pengguna ingin latihan prediksi harga saham berjalan lebih cepat, maka peneliti merekomendasikan pengguna untuk memenuhi spesifikasi rekomendasi perangkat keras ini. Berikut adalah spesifikasi rekomendasi perangkat keras :

- a. GPU dari NVIDIA dengan minimum VRAM sebesar 16GB dan arsitektur TURING.
- b. 16GB RAM.
- c. Minimum sisa penyimpanan sebesar 64GB.

1.2. Perancangan Penelitian

1.2.1. Desain Program

Sebelum program ini dibangun, terlebih dahulu dibuat sebuah desain dari program yang akan dibangun. Desain ini akan menjadi gambaran besar seperti apa program yang akan dibangun nantinya. Dari desain ini kita juga akan melihat peranan masing-masing komponen yang ada pada program.



Gambar 3.1

Desain Program Secara Sederhana

Secara sederhana, program yang dibangun nanti terdiri dari 5 bagian utama, yaitu :

- a. *Input data*
- b. *Data preprocessing.*
- c. Membuat Model LSTM
- d. *Training Model LSTM*
- e. *Output* dan Evaluasi hasil prediksi

Seperti yang sudah dijelaskan pada sub bab metode pengumpulan data, data perdagangan saham yang diambil dari Yahoo Finance akan menjadi data masukan yang digunakan. *Dataset*

ini kemudian diproses pada jaringan saraf atau *neural network* LSTM yang akan menghasilkan hasil prediksi harga saham berupa harga *closing* atau penutupan yang diinginkan.

Sebelum masuk pada tahapan proses membangun jaringan saraf atau *neural network* yang mampu memprediksi harga saham, terlebih dahulu peneliti menjelaskan parameter apa saja yang digunakan pada program ini, beberapa parameter jika diubah memiliki dampak yang besar terhadap akurasi prediksi harga saham, berikut adalah tabel berisi parameter dan penjelasan dari parameter tersebut :

TABEL 3.1
Parameter Yang Digunakan

Parameter	Fungsi
Ticker	Adalah parameter yang digunakan untuk menentukan saham yang diprediksikan, disini pengguna atau admin cukup memasukkan kode saham perusahaan, karena data saham diambil dari Yahoo! Finance yang mereka berbasis pasar saham New York, maka untuk kode saham perusahaan di Indonesia, pengguna atau admin harus menambahkan “.JK” diakhir kode saham, ini untuk memastikan bahwa saham yang diambil adalah saham yang berasal dari pasar saham IHSG.
EPOCHS	Adalah hyperparameter yang menggunakan nilai integer yang menentukan jumlah berapa kali program akan bekerja mengolah seluruh dataset latih, 1 epoch berarti setiap sampel dalam dataset latih memiliki 1 kali kesempatan untuk memperbarui parameter internal mereka.

LOOKUP_STEP	Adalah parameter dengan nilai integer yang menentukan hari prediksi, 1 berarti program akan melakukan prediksi pada 1 hari atau hari perdagangan berikutnya dari hari terakhir.
N_STEPS	Adalah parameter berupa nilai integer yang menentukan berapa banyak dataset latih yang digunakan untuk memprediksi harga saham kedepannya, nilai 100 berarti program akan menggunakan 100 hari perdagangan sebelumnya untuk menentukan prediksi harga saham berikutnya.
BATCH_SIZE	Adalah suatu nilai integer yang menentukan berapa banyak sampel yang diproses di dalam jaringan saraf atau neural network dalam satu waktu, disini peneliti menentukan nilai sebesar 64, yang berarti program akan memproses 64 sampel di dalam jaringan saraf dalam satu waktu.
N_LAYERS	Adalah sebuah nilai integer yang menentukan berapa banyak layer atau lapisan LSTM yang dibangun dan digunakan didalam jaringan saraf, disini peneliti menentukan bahwa 2 layer LSTM sudah cukup.
TEST_SIZE	Adalah sebuah nilai integer yang menentukan berapa persen data dibagi antara dataset latih dan dataset tes, disini peneliti menentukan 0.2 atau 20%, yang berarti 20% dari data yang dimasukkan akan dijadikan dataset tes.
BIDIRECTIONAL	Adalah sebuah nilai yang terdiri dari true atau false yang menentukan apakah jaringan saraf LSTM yang dibangun memiliki dua arah atau bidirectional atau tidak. Disini peneliti tidak menggunakan bidirectional yang artinya nilai parameter ini adalah false.

UNITS	Adalah sebuah nilai integer yang menentukan berapa banyak neuron LSTM yang dibangun dan digunakan pada model prediksi harga saham, disini peneliti menentukan nilai 256, yang berarti pada masing-masing layer LSTM memiliki 256 neuron.
OPTIMIZER	Optimizer adalah sebuah fungsi atau algoritma yang memodifikasi atribut jaringan saraf, seperti bobot dan kecepatan training. Dengan demikian, ini membantu dalam mengurangi loss keseluruhan dan meningkatkan akurasi.
LOSS	Adalah sebuah fungsi yang mengkalkulasikan jarak antara output saat ini dengan output yang diinginkan. Ini adalah metode untuk mengevaluasi bagaimana performa algoritma yang ada pada model data.
DROPOUT	Adalah sebuah teknik untuk menonaktifkan beberapa fungsi pada cell untuk mencegah terjadinya overfitting
SEQUENCE_LENGTH	Adalah nilai panjang sekuen data yang dimasukkan kedalam jaringan saraf
FEATURE_COLUMNS	Adalah kolom pada dataset yang digunakan untuk melatih dan memprediksi data

1.2.1.1. *Input* Data

Data saham yang sudah diambil dari API Yahoo_fin kemudian digunakan sebagai *input* pada program yang dibangun ini. *Input* ini kemudian dibagi menjadi 2 bagian, yaitu sebagai *dataset* latih atau *training* dan sebagai *dataset* *testing* atau tes.

1.2.1.2. Data *Preprocessing*

Ada beberapa tahap pada data *preprocessing* ini, berikut adalah beberapa tahapannya :

a. Memasukkan Data Kedalam Program

Data yang telah diambil dari Yahoo Finance kemudian dimasukkan kedalam program sebagai sebuah *dataframe*.

b. Menskalakan Data

Data yang dimasukkan kemudian diskalakan atau diubah dari format *integer* menjadi skala dari rentang 0 sampai 1.

c. Membangun *dataset* dan membagi *dataset*

Setelah data diskalakan menjadi rentang 0 sampai 1, kemudian data dimasukkan kedalam 2 *dataframe* X dan Y kemudian dijadikan sebuah *dataset* X_latih, X_test, Z_latih, Z_test.

1.2.1.3. Membuat Model LSTM

Sebelum memulai proses *training* atau latihan, sebelumnya dilakukan proses pembuatan model prediksi, model yang dibuat ini bersifat *global*, yang artinya model dapat dipanggil pada proses lainnya, pembuatan model ini menggunakan beberapa parameter yang sudah dijelaskan pada tabel 3.1, beberapa parameter yang digunakan adalah N_LAYERS, UNITS, CELL, DROPOUT, LOSS, OPTIMIZER, BIDIRECTIONAL, N_FEATURES, SEQUENCE_LENGTH.

1.2.1.4. *Training* Model LSTM

Setelah membuat model, selanjutnya adalah tahap *training* atau latihan model LSTM, latihan ini bertujuan agar model memahami mengenai data yang akan diprediksi, dengan melatih model LSTM maka model dapat mengetahui pola-pola pergerakan data dalam rentang waktu tertentu, dan dengan latihan yang baik maka kemungkinan besar prediksi yang dihasilkan semakin tinggi tingkat akurasinya, untuk melatih model kedua *dataset* dimasukkan kedalam fungsi latihan yang disebut dengan *fitting*, pada fungsi ini digunakan 2 parameter utama, yaitu EPOCHS dan BATCH_SIZE, dengan memodifikasi kedua nilai parameter dengan nilai yang tinggi, maka kemungkinan besar model prediksi memiliki tingkat akurasi tinggi, namun tingginya parameter latihan akan membuat proses latihan lebih berat untuk *hardware* yang digunakan, jika spesifikasi dan kinerja *hardware* tidak terlalu tinggi, maka proses latihan memakan waktu yang cukup lama, setelah model dilatih, maka hasil terbaik dari proses latihan disimpan, yang kemudian model hasil latihan tersebut digunakan untuk memprediksi harga saham.

1.2.1.5. *Output* Dan Evaluasi Hasil Prediksi

Pada proses ini digunakan beberapa parameter diantaranya adalah N_STEPS, LOOKUP_STEP, LOSS, CELL, N_LAYERS, OPTIMIZER, DROPOUT, BIDIRECTIONAL, FEATURE_COLUMNS. Evaluasi model dilakukan pada model latihan dan model prediksi, pada model latihan peneliti menggunakan fungsi *loss* yaitu MAE (*Mean Absolute Error*) untuk menentukan

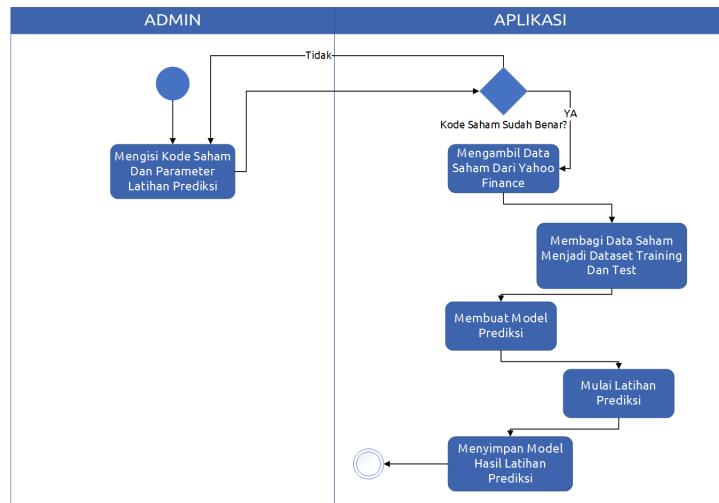
seberapa besar nilai MAE yang didapatkan, seperti yang telah dijelaskan pada subab sebelumnya, bahwa semakin kecil nilai MAE maka semakin besar kemungkinan model memiliki akurasi yang tinggi, untuk hasil prediksi, peneliti menentukan tingkat akurasi dengan membandingkan hasil akurasi dengan harga saham sebenarnya.

1.2.2. *Activity Diagram*

Pada bagian ini peneliti menjelaskan proses-proses yang terjadi pada program atau aplikasi prediksi harga saham, terdapat 2 proses yaitu melatih model prediksi dan melihat hasil prediksi, melatih model prediksi adalah kegiatan yang dilakukan oleh admin, sedangkan pengguna hanya bisa melihat hasil prediksi, perlu diperhatikan bahwa kedua proses ini adalah proses saat aplikasi dijalankan menggunakan Google Colab.

1.2.2.1. Proses Latihan Model Prediksi

Proses ini adalah proses dimana model prediksi dilatih agar dapat memprediksi harga saham dengan akurasi terbaik yang bisa didapatkan, proses ini dimulai dari admin memasukkan kode saham yang ingin diprediksi pada *input* yang tersedia, kemudian memasukkan beberapa parameter prediksi.



Gambar 3.2

Activity Diagram Melatih Model Prediksi

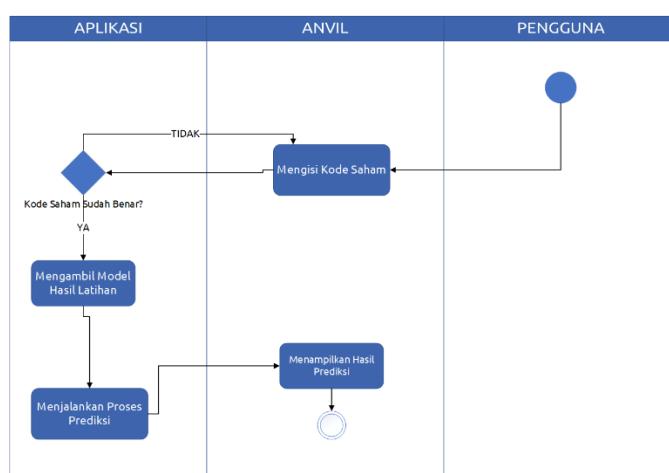
Dapat dilihat pada gambar 3.2 adalah *activity diagram* dari proses latihan model prediksi, proses ini dilakukan oleh *admin*. Berikut

- Admin* memasukkan kode saham dan parameter, kemudian kode saham dan parameter ini masuk ke sistem aplikasi.
- Selanjutnya aplikasi menentukan apakah kode saham sudah sesuai pada Yahoo Finance dengan menggunakan API Yahoo Finance, jika kode saham telah sesuai maka selanjutnya adalah mengambil data perdagangan saham sesuai dengan data saham tersebut.
- Setelah data perdagangan saham berhasil diambil, maka data dibagi menjadi dua *dataset* pembagian ini ditentukan oleh parameter TEST_SIZE dalam format persentase, selanjutnya adalah membuat model prediksi berdasarkan parameter yang ada.
- Setelah model dibuat maka kemudian adalah menjalankan latihan model prediksi.

- e. Setelah proses latihan selesai, maka versi model terbaik disimpan.

1.2.2.2. Melihat Hasil Prediksi

Setelah melihat proses latihan model prediksi, kemudian adalah melihat hasil prediksinya, proses ini dilakukan oleh pengguna, pengguna cukup memasukkan kode saham yang telah dilatih untuk melihat hasil prediksi, proses ini merupakan proses yang terdapat *client-side*, dimana pada Anvil tersedia sebuah form yang digunakan oleh pengguna untuk melihat hasil prediksi.



Gambar 3.3

Activity Diagram Menampilkan Hasil Prediksi

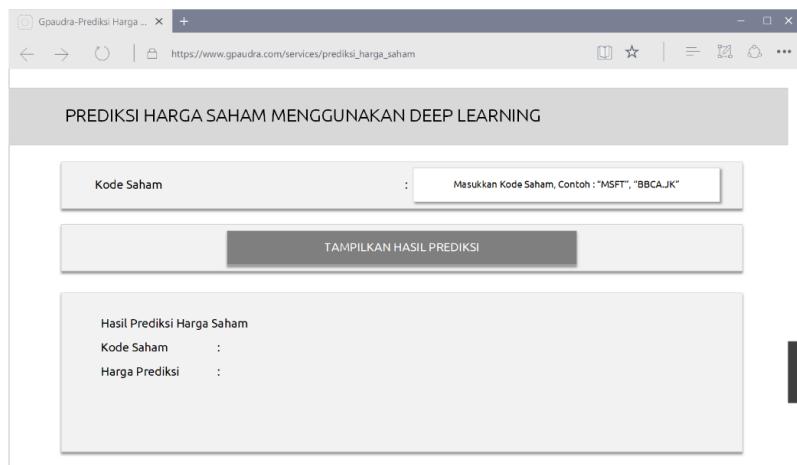
Dapat dilihat pada gambar 3.2 adalah *activity diagram* dari proses menampilkan hasil prediksi, proses ini secara spesifik berikut adalah penjelasan dari proses yang terjadi :

- a. Pengguna memasukkan kode saham pada *form* yang ada di Anvil
- b. Kemudian Anvil mengirim kode saham ke aplikasi dan aplikasi mengecek apakah kode saham sudah sesuai

dengan kode yang ada pada Yahoo Finance melalui API Yahoo Finance.

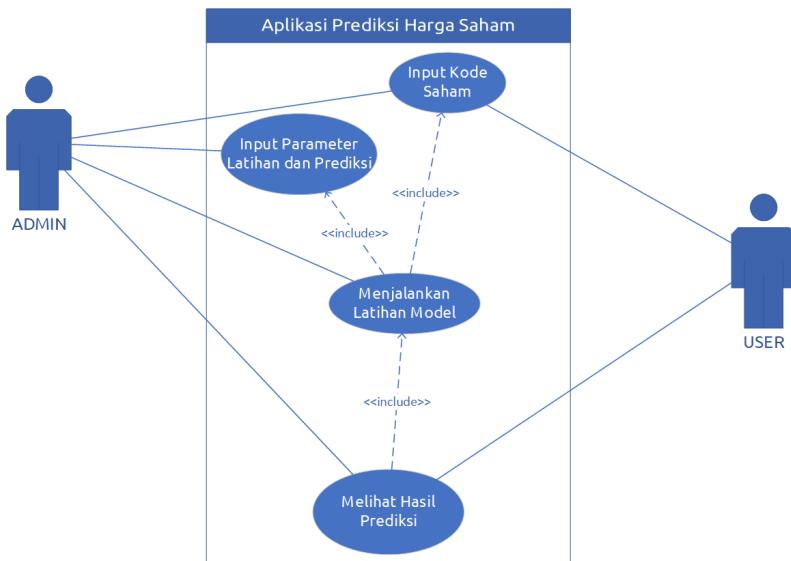
- c. Jika kode saham sesuai maka kemudian aplikasi mengambil model hasil latihan.
- d. Kemudian proses prediksi dijalankan.
- e. Setelah proses prediksi selesai maka hasil prediksi dikirim ke Anvil agar dapat dilihat oleh pengguna.

1.2.3. Pemodelan *User Interface*



Gambar 3.4
Desain *User Interface* Untuk *Client-Side*

1.2.4. Use-case Diagram



Gambar 3.5

Use-case Diagram Aplikasi Prediksi Harga Saham

Dapat dilihat pada gambar 3.5 adalah diagram *use-case* dari aplikasi prediksi harga saham, disini terdapat dua aktor, yaitu *admin* dan *user* atau pengguna, kedua aktor memiliki 2 interaksi yang sama pada aplikasi, yaitu dapat *input* kode saham dan melihat hasil prediksi, namun untuk *input* parameter dan menjalankan latihan model prediksi hanya admin yang bisa berinteraksi dengan kedua fungsi tersebut, untuk menjalankan fungsi melihat hasil prediksi, dibutuhkan fungsi latihan model prediksi, fungsi latihan model prediksi juga membutuhkan fungsi parameter dan kode saham agar latihan model dapat dijalankan.

1.3. Teknik Analis

Teknik analis adalah suatu metode yang digunakan untuk mengolah data menjadi informasi yang berguna, atau dalam arti lain adalah sebuah kegiatan untuk mengubah data yang dihasilkan oleh sebuah atau beberapa penelitian yang telah dilakukan menjadi informasi yang berguna dan dapat digunakan untuk mengambil kesimpulan.

Pada bagian ini peneliti mencoba melakukan perhitungan algoritma LSTM, MAE, dan akurasi prediksi.

a. Perhitungan LSTM

TABEL 3.2
Nilai Input LSTM

Data 1	Data 2
0.70	0.90
...	...

Kemudian dimisalkan nilai bobot atau *weight* yang didapatkan adalah sebagai berikut :

$$U_f = [0.77, 0.55], U_i = [0.88, 0.67], U_o = [0.57, 0.70],$$

$$U_c = [0.50, 0.35]$$

$$W_f = 0.03, W_i = 0.50, W_c = 0.30, W_o = 0.02$$

$$b_f = 0.15, b_i = 0.01, b_c = 0.05, b_o = 0.10$$

$$h_{t-1} = 0, C_{t-1} = 0$$

1) *Forget Gate* :

$$f_t = \sigma(U_f X_t + W_f h_{t-1} + b_f) \quad (4)$$

$$f_t = \sigma((0.77 * 0.70) + (0.55 * 0.90)) + 0.03 * 0 + 0.15$$

$$f_t = \sigma(1.034 + 0.15)$$

$$f_t = \sigma(1.184)$$

$$f_t = \frac{1}{1 + e^{-1.184}}$$

$$f_t = 0.765$$

2) *Input Gate* :

$$i_t = \sigma(U_i X_t + W_i h_{t-1} + b_i) \quad (5)$$

$$\begin{aligned}
i_t &= \sigma((0.88 * 0.70) + (0.67 * 0.90)) + 0.5 * 0 + 0.01 \\
i_t &= \sigma(1.219 + 0.01) \\
i_t &= \sigma(1.229) \\
i_t &= \frac{1}{1 + e^{-1.229}} \\
i_t &= 0.773
\end{aligned}$$

3) *Cell State*

$$\begin{aligned}
\tilde{C}_t &= \tanh(U_c X_t + W_c h_{t-1} + b_c) & (6) \\
\tilde{C}_t &= \tanh(((0.50 * 0.70) + (0.35 * 0.90)) + 0.3 * 0 + 0.05) \\
\tilde{C}_t &= \tanh(0.665 + 0.05) \\
\tilde{C}_t &= \tanh(0.715) \\
\tilde{C}_t &= \frac{e^{0.715} - e^{-0.715}}{e^{0.715} + e^{-0.715}} \\
\tilde{C}_t &= 0.613 \\
C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t & (7) \\
C_t &= (0.765 \odot 0 + 0.773 \odot 0.613) \\
C_t &= 0.473
\end{aligned}$$

4) *Output Gate* :

$$\begin{aligned}
O_t &= \sigma(U_o X_t + W_o h_{t-1} + b_o) & (8) \\
O_t &= \sigma((0.57 * 0.70) + (0.70 * 0.90)) + 0.002 * 0 + 0.10 \\
O_t &= \sigma(1.029 + 0.10) \\
O_t &= \sigma(0.129) \\
O_t &= \frac{1}{1 + e^{-0.129}} \\
O_t &= 0.532 \\
h_t &= O_t * \tanh(C_t) & (9) \\
h_t &= O_t * \tanh(0.473) \\
h_t &= 0.532 * \frac{e^{0.473} - e^{-0.473}}{e^{0.473} + e^{-0.473}} \\
h_t &= 0.532 * 0.440
\end{aligned}$$

$$h_t = 0.234$$

b. *Mean Absolute Error :*

Misal : $n = 7, F^i = 0.97, A^i = 0.70$

$$MAE = \frac{1}{n} \sum_{i=1}^n |A^i - F^i| \quad (10)$$

$$MAE = \frac{1}{7} \sum_{i=1}^7 |0.70 - 0.97|$$

$$MAE = \frac{1}{7} 1.89$$

$$MAE = 0.27$$

c. Nilai Akurasi Prediksi

Misal : $N_p = 7700, N_a = 8000$

$$A = 1 - \left| \frac{N_p}{N_a} \right| \times 100\% \quad (11)$$

$$A = 1 - \left| \frac{7700}{8000 - 1} \right| \times 100\%$$

$$A = 0.962 \times 100\%$$

$$A = 96.2\%$$

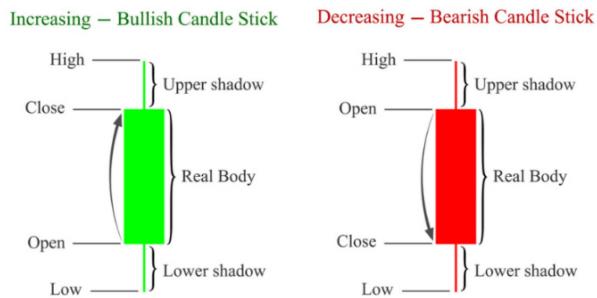
d. Perbandingan Prediksi Manual

Prediksi saham secara manual bisa dilakukan melalui analisis *candlestick chart* atau grafik lilin, grafik ini digunakan oleh para *trader* sebagai rambu-rambu dalam perdagangan saham, grafik ini cocok untuk memprakirakan nilai atau harga saham pada hari saat perdagangan terjadi dan hari esok.



Gambar 3.6
Contoh *Candlestick Chart*
Sumber : (Awal, 2022)

Komponen pada grafik *candlestick* memuat empat harga, yaitu harga pembukaan atau *open*, penutupan atau *close*, tertinggi atau *high*, dan terendah atau *low*, komponen *candlestick* terdiri dari badan atau *body*, dan ekor atau *shadow*.



Gambar 3.7
Komponen *Candlestick*
Sumber : (Awal, 2022)

Ukuran badan atau *body* pada *candlestick* menunjukkan seberapa besar pergerakan harga saham saat perdagangan terjadi, jika ukuran badan panjang berarti momentum perdagangan sedang menguat alias banyak transaksi perdagangan yang terjadi. Jika *candlestick* berwarna merah maka itu berarti harga penutupan lebih

rendah dari harga pembukaan, sedangkan hijau berarti harga penutupan lebih tinggi dari harga pembukaan, kemudian jika ekor *candlestick* memanjang kebawah itu berarti harga saham dipaksa untuk turun, sedangkan panjang keatas berarti harga saham dipaksa untuk naik oleh para *trader*.

Untuk mengetahui apakah saham akan naik atau turun dengan *candlestick*, para *trader* menggunakan beberapa pola, diantaranya :

1) Pola *Doji*



Gambar 3.8

Pola *Doji* Pada Saham INCO

Sumber : (Awal, 2022)

Pola *Doji* pada Gambar 3.8 memberikan sinyal akan adanya tren penurunan saham setelah kenaikan saham, jika pola ini ada ini adalah sinyal bagi para *trader* dan investor untuk menjual saham mereka.

2) Pola Hammer



Gambar 3.9

Pola Hammer Pada Saham INDF

Sumber : (Awal, 2022)

Pola *hammer* pada Gambar 3.9 menunjukkan akan terjadinya tren kenaikan harga saham, ini adalah sinyal bagi para *trader* dan investor untuk membeli atau mempertahankan saham tersebut.

3) Pola Marubozu



Gambar 3.10

Pola Marubozu Pada Saham ITMG

Sumber : (Awal, 2022)

Pola *Marubozu* dibagi menjadi dua, yaitu berwarna hijau dan merah, apabila *marubozu* berwarna hijau ini berarti saham

akan mengalami tren kenaikan, jika merah seperti Gambar 3.10, maka saham akan mengalami tren penurunan.

Dari beberapa contoh yang telah diberikan, dapat dilihat bahwa dengan memahami pola-pola yang ada pada grafik *candlestick* investor maupun *trader* dapat memprediksi naik atau turunnya harga suatu saham, namun hal ini tidak bisa diukur tingkat akurasi dan tingkat seberapa besar prediksi pergerakan saham yang akan terjadi, dan juga jika dikemudian hari keadaan ekonomi maupun sosial berubah secara drastis maka pola-pola pada grafik menjadi tidak berguna.

Berbeda dengan menggunakan suatu aplikasi, hasil prediksi dengan harga sebenarnya dapat diukur tingkat akurasinya, dan juga dengan menggunakan aplikasi atau program kita bisa melihat prediksi lebih jauh kedepan dengan tingkat akurasi yang masih cukup tinggi, besarnya tingkat akurasi yang didapat dengan menggunakan aplikasi prediksi harga saham sudah peneliti bahas pada Bab IV penelitian ini, namun sama halnya dengan prediksi manual yang telah disebutkan, aplikasi ini tidak menggunakan data dari sentimen atau pengaruh keadaan sosial dan ekonomi, sehingga jika keadaan ekonomi dan atau sosial yang berubah drastis keesokan atau beberapa hari berikutnya, maka hasil prediksi kemungkinan besar memiliki tingkat akurasi yang rendah.

BAB IV

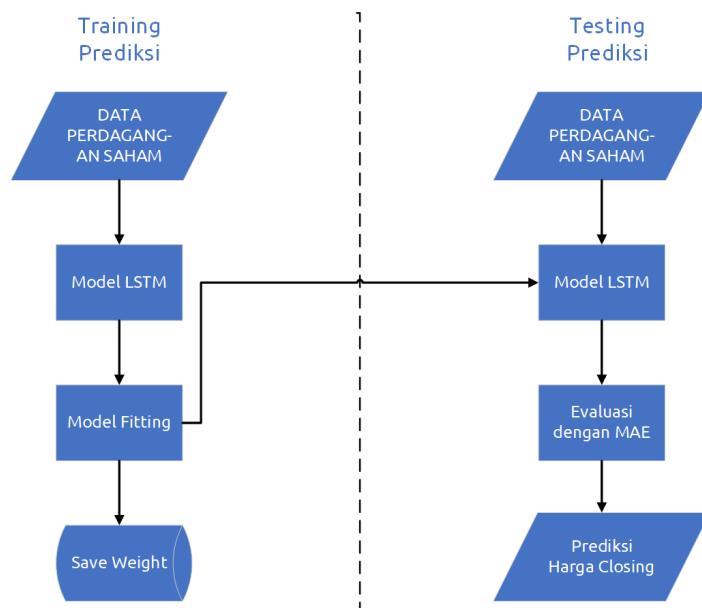
HASIL DAN PEMBAHASAN

4.1. Hasil

4.1.1. Implementasi LSTM

Sebelum LSTM diimplementasikan untuk memprediksi harga saham, perlu dibuatnya model jaringan saraf LSTM dan melatih atau *training* jaringan saraf LSTM ini sehingga model ini dapat mengenali dan paham akan tugasnya yaitu memprediksi harga *closing* saham. Pada tahap latihan atau *training*, terdapat 2 model, yaitu model *training* dan model *testing*.

4.1.1.1. Tahap *Training*



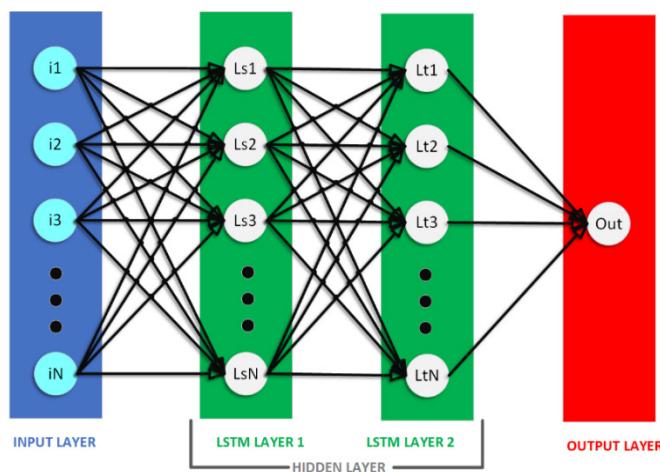
Gambar 4.1

Skema latihan prediksi harga saham

Dalam program prediksi harga saham ini, untuk mendapatkan akurasi prediksi yang baik, maka LSTM perlu dilatih dengan sejumlah data *training*. Tujuan dari melatih LSTM adalah untuk

menemukan ciri dari setiap data yang diberikan kemudian menandai *neuron-neuron* yang ada pada *neural network* atau jaringan saraf, oleh karena itu perlu dibuatnya sebuah model untuk melakukan latihan agar saat nanti dilakukan pengujian untuk melakukan prediksi, maka program sudah terlatih dengan baik.

Sebelum memulai latihan, perlu dibuatnya sebuah model LSTM, disini peneliti sudah membuat visualisasi bagaimana model LSTM tersusun.



Gambar 4.2

Visualisasi *layers* yang digunakan pada model LSTM

Pada gambar diatas, dapat dilihat bahwa model LSTM yang peneliti bangun memiliki 4 *layers* atau lapisan, mulai dari kiri yaitu terdapat lapisan pertama yang disebut dengan *input layer*, pada lapisan ini data dari *dataset* yang sudah dibentuk dimasukkan, dapat dilihat bahwa lapisan ini terdiri dari *cell* i_1 sampai dengan *cell* i_N , jumlah *cell* ini tergantung dari berapa banyak data perdagangan saham yang ditentukan oleh admin, misalnya disini peneliti memasukkan nilai 80, berarti program akan menggunakan 80 hari terakhir

data perdagangan sebagai *dataset* yang digunakan untuk melatih model prediksi.

Kemudian lapisan berikutnya adalah LSTM *layer 1* dan LSTM *layer 2* yang keduanya termasuk *hidden layer* atau lapisan tersembunyi, pada rancangan ini peneliti hanya menggunakan 2 lapisan LSTM, kedua lapisan LSTM terdiri dari *cell 1* sampai dengan *cell N*, jumlah *cell* ini dapat peneliti ubah pada parameter UNITS. Model LSTM ini peneliti bangun dalam sebuah *function* atau fungsi.

Setelah model dibentuk, terakhir adalah *compile* atau kompilasi *code* agar model dapat dieksekusi, disini parameter yang digunakan adalah LOSS, METRICS, OPTIMIZER. Tahap selanjutnya setelah membuat model LSTM adalah membuat model fitting yang digunakan untuk latihan atau *training* prediksi harga saham, untuk membuat model *fitting*, diperlukan beberapa parameter, seperti EPOCHS, BATCH_SIZE, verbose, dan data yang akan dilatih. Disaat menjalankan kode yang terdapat model *fitting*, maka proses *training* langsung dilakukan, lamanya waktu *training* tergantung dari spesifikasi *hardware* yang digunakan.

4.1.1.2. Tahap Evaluasi Dan *Testing*

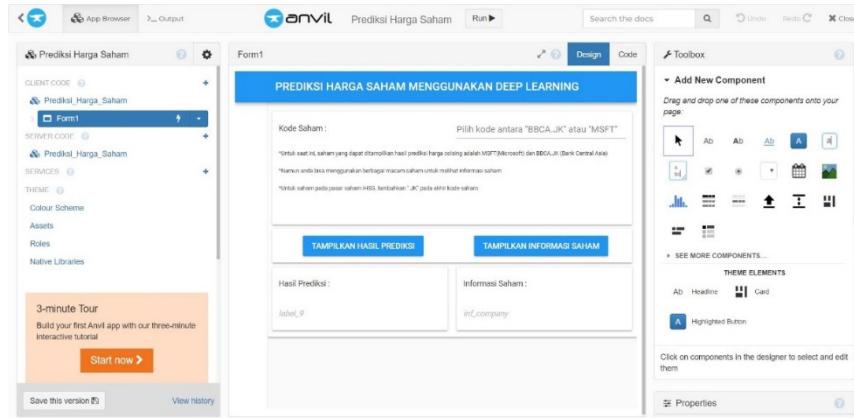
Setelah membuat model *fitting*, kemudian membuat model prediksi atau *testing*, proses testing ini memerlukan model *training* yang sudah dilatih pada tahap *fitting* sebelumnya, selain menghasilkan nilai prediksi, disini peneliti juga membuat sebuah grafik plot

yang berisi perbandingan nilai asli dari harga *closing* saham dan nilai prediksi harga *closing* saham mulai dari IPO sampai perdagangan terakhir.

4.1.2. Menampilkan Hasil Prediksi Pada *Client-side*

Setelah melihat proses membangun model LSTM untuk latihan dan testing prediksi harga saham, kemudian pada sub bab ini peneliti menjelaskan bagaimana untuk menampilkan atau mengirim data dari Google Colab atau Python Notebook atau *backend* ke sebuah client side, client side yang digunakan peneliti adalah Anvil Works, Anvil Works sudah peneliti jelaskan pada Bab 2 penelitian ini, tujuan untuk menampilkan harga saham pada client side adalah untuk memudahkan pengguna awam yang ingin mencoba melihat prediksi harga saham, sebelum melihat bagaimana untuk memanggil data ke Anvil, diperlukan pememasang modul Anvil pada Python *Environment*, untuk memasang Anvil cukup mudah, yaitu dengan menggunakan perintah berikut `pip install anvil-uplink` pada *command line python environment*.

Setelah modul anvil berhasil dipasang, maka selanjutnya adalah menghubungkan *notebook* dengan *client-side* pada Anvil, kode *uplink* bisa didapatkan pada menu pengaturan aplikasi, kemudian pilih *uplink* dan akan diperlihatkan kode *uplink* beserta kode python untuk menghubungkan *notebook/backend* dengan *client side*. Setelah berhasil menghubungkan ke *client side*, selanjutnya adalah bagaimana untuk membuat sebuah *function* untuk menampilkan hasil prediksi, sebelum itu harus dibuat UI pada Anvil Works.



Gambar 4.3

Contoh pembuatan UI *client-side* pada Anvil Works

Pembaca dapat mengikuti desain yang sama yang telah peneliti buat, dan pembaca juga bisa membuat yang berbeda, yang terpenting adalah terdapat sebuah kolom atau *textbox input* yang dapat digunakan oleh pengguna untuk memasukkan kode saham yang dipilih, sebuah tombol, dan *output* yang menampilkan hasil prediksi.

Untuk memulai membuat fungsi, dapat memilih tombol untuk menampilkan hasil prediksi, kemudian pada menu *properties* yang ada di bagian kanan, *scroll* kebawah sehingga terdapat tanda panah biru ke kanan, dan klik panah yang sejajar dengan tulisan *click*, kemudian akan diarahkan ke halaman *code* pada Anvil, dengan menekan tombol panah tersebut maka Anvil secara otomatis membuat sebuah *function* yang jika pengguna menekan tombol tersebut, maka suatu *event* akan terjadi.

Kemudian masuk pada *function* yang telah dibuat tersebut, didalam *function* ini peneliti membuat sebuah variabel yang berisi *method* untuk mengambil data dari program prediksi, disini peneliti membuat sebuah tombol yang memiliki *click function* yaitu jika tombol tersebut diklik maka akan mengirim nilai yang dimasukkan

oleh pengguna ke *backend* aplikasi prediksi harga saham, kemudian hasil prediksi dikembalikan atau *return* ke *client-side* pada Anvil.

Tahap berikutnya adalah yang ada pada program prediksi harga saham atau *backend* atau *python notebook*, disini peneliti membuat sebuah *function* yang sesuai dengan *function* yang ada pada *client side* atau aplikasi Anvil, yang diberi nama *function* ‘`kirimdatakeanvil`’. Sebelum mendeklarasikan *function* untuk mengirim data hasil prediksi, perlu ditambahkan sebuah *method* yaitu `@anvil.server.callable` yang mengindikasikan bahwa *function* dibawah *method* tersebut adalah *function* yang digunakan untuk *client side* pada Anvil. Variabel `hasilsaham` adalah sebuah variabel yang dikirim ke Anvil sebagai *output* dari prediksi harga saham. Perlu diingat kembali bahwa *source code* yang peneliti buat ini adalah *source code* pada Google Colab yang menggunakan Google Drive sebagai penyimpanan model hasil latihan prediksi dan hasil prediksi harga saham.

4.2. Pembahasan

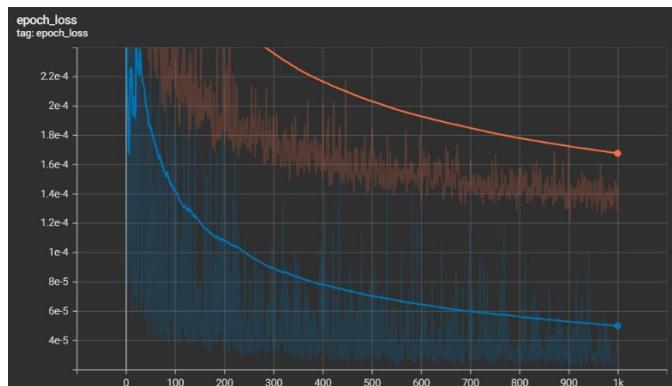
4.2.1. Hasil Proses *Training*

Seperti yang sudah dijelaskan pada kedua bab sebelumnya, bahwa setiap jaringan saraf atau *neural network* membutuhkan sebuah proses pelatihan atau *training* agar model dapat mengetahui pola-pola data dan membantu memprediksi data dimasa mendatang. Proses *training* yang telah dilakukan menggunakan 800 dan 1000 *epochs* pada data saham BBCA dan 1000 *epochs* pada data saham MSFT.

TABEL 4.1
Parameter Latihan Model LSTM

Parameter	Nilai
BATCH_SIZE	64
EPOCHS	1000

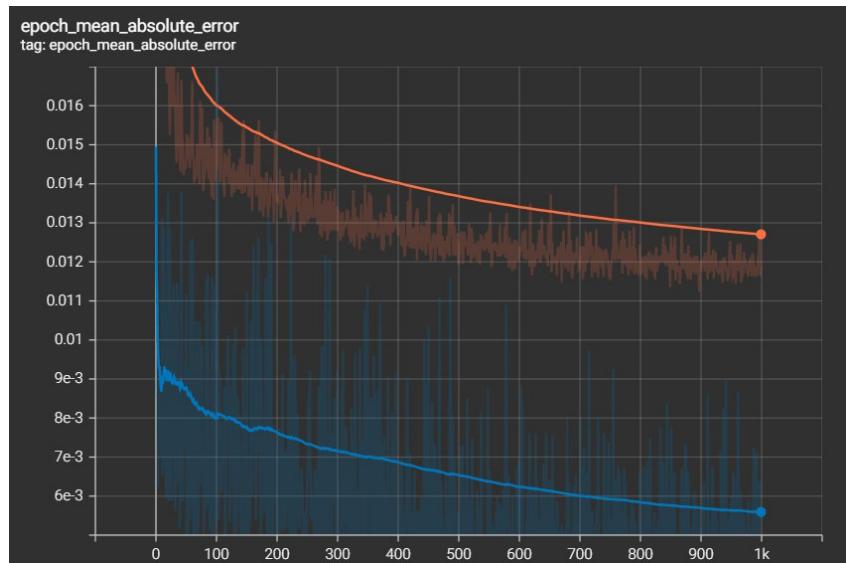
Tabel diatas adalah parameter yang peneliti gunakan untuk melatih model LSTM prediksi harga saham, dengan menggunakan parameter tersebut maka model LSTM memiliki nilai *loss* dan MAE yang rendah, terlihat bahwa BATCH_SIZE bernilai 64, yang artinya sebanyak 64 sampel data secara serentak diproses pada jaringan saraf didalam model *training*, kemudian adalah EPOCHS yang bernilai 1000, yang artinya setiap parameter pada jaringan saraf didalam model LSTM diperbarui sebanyak 1000 kali.



Gambar 4.4

Grafik nilai *loss* pada hasil *training* dan validasi dengan 1000 *epochs*

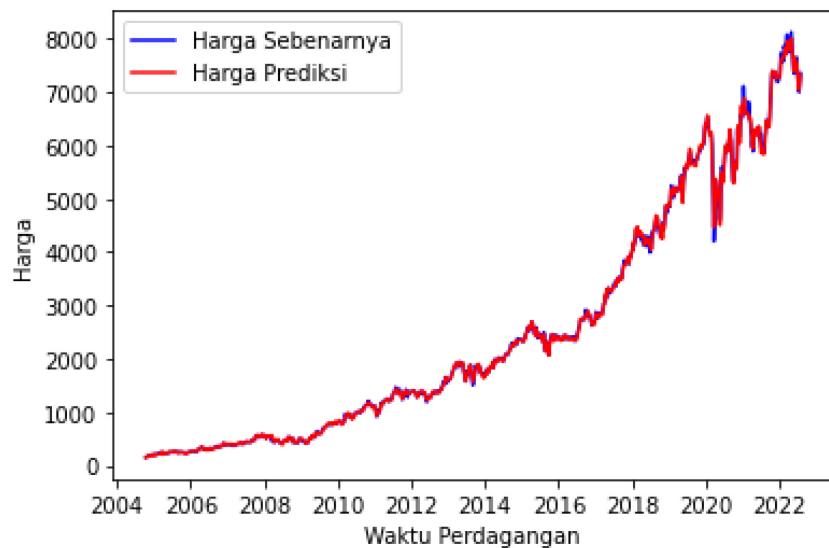
Dapat dilihat pada gambar 4.11 adalah nilai *loss* terus menurun seiring dengan meningkatnya jumlah *epochs*, ini membuktikan pada *training* yang telah peneliti lakukan bahwa semakin banyak *epochs* maka semakin kecil nilai *loss* pada model, yang berarti membuat besar kemungkinan tingkat akurasi semakin tinggi, garis *orange* pada grafik tersebut adalah *loss* dari hasil *training* saham BBCA yang memiliki nilai loss sebesar 0,000019.



Gambar 4.5

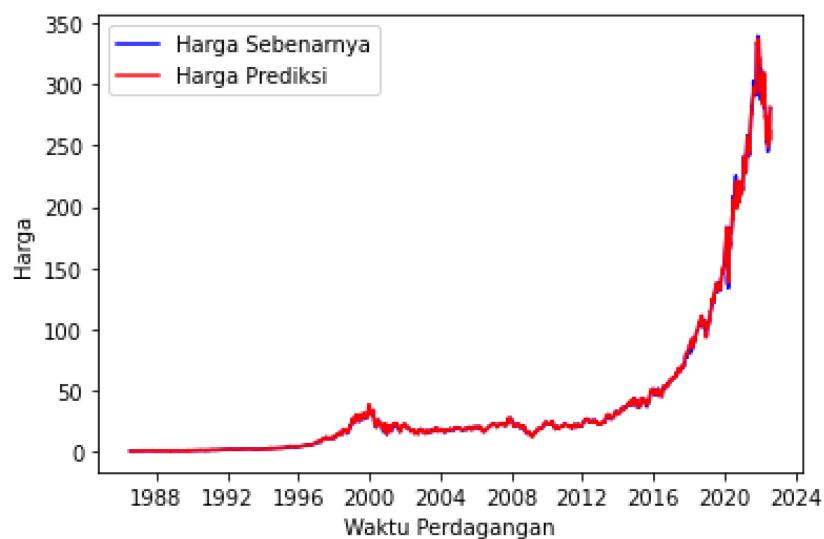
Grafik nilai MAE pada proses *training* dan validasi dengan 1000
epochs

Kemudian adalah nilai dari MAE atau *Mean Absolute Error*, gambar diatas adalah grafik dari nilai MAE pada proses *training* dan validasi model, dapat dilihat bahwa nilai MAE terus menurun seiring bertambahnya *epochs*, ini membuktikan bahwa semakin banyak *epochs* maka semakin berkurang nilai MAE yang membuat model semakin akurat, garis *orange* pada grafik tersebut adalah nilai MAE pada model saham BBCA memiliki nilai MAE sebesar 0.0051.



Gambar 4.6

Grafik *plot* perbandingan hasil prediksi dan harga sebenarnya pada saham BBCA



Gambar 4.7

Grafik *plot* perbandingan hasil prediksi dan harga sebenarnya pada saham MSFT

Dapat dilihat pada gambar 4.13 dan 4.14, keduanya merupakan grafik *plot* perbandingan harga prediksi dan harga sebenarnya yang dilakukan oleh aplikasi setelah proses latihan

dilakukan, kedua grafik ini merupakan salah satu visualisasi dari hasil latihan mengenai bagaimana tingkat akurasi model setelah dilatih. Waktu yang dibutuhkan untuk menyelesaikan proses *training* sebanyak 1000 *epochs* melalui Google Colab versi gratis bervariasi, paling cepat adalah 1,17 detik per *epochs* atau 1170 detik, yang artinya proses *training* prediksi harga saham sebanyak 1000 *epochs* pada Google Colab yang tercepat berlangsung selama 19,5 menit, untuk proses terlama yang peneliti alami mencapai 2,19 detik per *epochs* atau sekitar 36,5 menit, peneliti tidak bisa mengkonfirmasi secara pasti bagaimana bisa terjadi variasi waktu latihan yang berbeda dua kali lipat, ini dikarenakan pihak Google tidak memberikan informasi apapun mengenai hal tersebut.

4.2.2. Hasil Prediksi

Pada bagian ini peneliti menampilkan hasil prediksi yang telah peneliti lakukan, hasil prediksi disini yang pertama adalah prediksi yang dilakukan secara harian, yang berarti peneliti melakukan proses *training* dan *testing* setiap hari selama 7 hari untuk melihat akurasi model prediksi harga *closing* saham dengan harga *closing* sebenarnya, yang kedua adalah hasil prediksi dalam rentang waktu yang berbeda, yang artinya peneliti menentukan *target* hari prediksi secara berbeda-beda namun tetap menggunakan data latih dan model latihan yang sama, saham yang digunakan disni adalah saham MSFT dari perusahaan Microsoft dan saham kedua adalah saham BBCA dari perusahaan Bank Central Asia atau BCA.

TABEL 4.2
Parameter Prediksi Model LSTM

Parameter	Nilai
N_STEPS	80
N_LAYERS	2
LOSS	mean_absolute_error
UNITS	256
DROPOUT	0,3
OPTIMIZER	rmsprop
BIDIRECTIONAL	FALSE
CELL	LSTM

Tabel 4.2 adalah parameter yang peneliti gunakan pada model LSTM untuk menghasilkan prediksi harga saham pada 1 hari setelah data saham terakhir, dapat dilihat bahwa parameter N_STEPS bernilai 80, yang artinya peneliti menentukan bahwa model menggunakan 80 hari terakhir data saham perdagangan sebagai acuan untuk memprediksi harga saham 1 hari berikutnya, kemudian parameter N_LAYERS bernilai 2 yang artinya peneliti menentukan bahwa model prediksi memiliki 2 *layer* LSTM pada model prediksi, kemudian menggunakan MAE sebagai parameter *loss*, dan UNITS bernilai 256 yang artinya terdapat 256 *cell* LSTM pada model prediksi, kemudian DROPOUT adalah 0,3 yang berarti peneliti menentukan 30% maksimum *dropout* pada model.

TABEL 4.3
Hasil prediksi harga saham MSFT secara harian

TANGGAL DATA SAHAM TERAKHIR	PREDIKSI PADA HARI KE-X	TANGGAL YANG PREDIKSIKAN	HARGA CLOSING SEBELUM-NYA	PREDIKSI HARGA CLOSING	HARGA CLOSING REAL	PREDIKSI PERGERAKAN	PERGERAKAN REAL	NILAI AKURASI PREDIKSI
13 Juli 2022	1	14 Juli 2022	\$252,72	\$254,11	\$254,08	▲0,55%	▲0,54%	100,0%
14 Juli 2022	1	15 Juli 2022	\$254,08	\$252,80	\$256,72	▼0,51%	▲1,03%	98,5%
15 Juli 2022	1	16 Juli 2022	\$256,72	\$253,98	\$254,25	▼1,08%	▼0,97%	99,9%
18 Juli 2022	1	19 Juli 2022	\$254,25	\$259,21	\$259,53	▲1,91%	▲2,03%	99,9%
19 Juli 2022	1	20 Juli 2022	\$259,53	\$259,16	\$262,27	▼0,14%	▲1,04%	98,8%
20 Juli 2022	1	21 Juli 2022	\$262,27	\$261,00	\$264,84	▼0,49%	▲0,97%	98,6%
21 Juli 2022	1	22 Juli 2022	\$264,84	\$263,15	\$260,36	▼0,64%	▼1,72%	98,9%
							Rata-Rata :	99,2%

TABEL 4.4
Hasil prediksi harga saham BBCA secara harian

TANGGAL DATA SAHAM	PREDIKSI PADA HARI KE-X	TANGGAL YANG PREDIKSIKAN	HARGA CLOSING SEBELUM-NYA	PREDIKSI HARGA CLOSING	HARGA CLOSING REAL	PREDIKSI PERGERAKAN	PERGERAKAN REAL	NILAI AKURASI PREDIKSI
14 Juli 2022	1	15 Juli 2022	Rp7.025,00	Rp7.023,00	Rp7.000,00	▼0,03%	▼0,36%	99,7%
15 Juli 2022	1	18 Juli 2022	Rp7.000,00	Rp7.093,00	Rp7.150,00	▲1,31%	▲2,10%	99,2%
18 Juli 2022	1	19 Juli 2022	Rp7.150,00	Rp7.045,00	Rp7.175,00	▼1,49%	▲0,35%	98,2%
19 Juli 2022	1	20 Juli 2022	Rp7.175,00	Rp7.067,77	Rp7.400,00	▼1,52%	▲3,04%	95,5%
20 Juli 2022	1	21 Juli 2022	Rp7.400,00	Rp7.301,04	Rp7.400,00	▼1,36%	►0,00%	98,7%
21 Juli 2022	1	22 Juli 2022	Rp7.400,00	Rp7.339,76	Rp7.325,00	▼0,82%	▼1,02%	99,8%
22 Juli 2022	1	25 Juli 2022	Rp7.325,00	Rp7.286,27	Rp7.300,00	▼0,53%	▼0,34%	99,8%
							Rata-Rata	98,7%

Dapat dilihat pada kedua tabel diatas adalah hasil dari prediksi harga saham, kedua tabel memuat data mengenai harga prediksi *closing*, harga *closing* sebenarnya, prediksi pergerakan harga, dan pergerakan harga sebenarnya, kemudian yang terakhir adalah nilai akurasi prediksi, seperti yang telah dijelaskan pada bab 3 subab teknik analisis, bahwa nilai akurasi prediksi adalah sebuah nilai yang menjadi salah satu nilai penentu dari penelitian ini.

TABEL 4.5
Hasil Prediksi Harga Saham BBCA Dalam Rentang Waktu
Berbeda

TANGGAL DATA SAHAM TERAKHIR	PREDIKSI PADA HARI KE-X	TANGGAL YANG PREDIKSIKAN	HARGA CLOSING SEBELUM-NYA	PREDIKSI HARGA CLOSING	HARGA CLOSING REAL	PREDIKSI PERGERAKAN	PERGERAKAN REAL	NILAI AKURASI PREDIKSI
11 Juli 2022	1	12 Juli 2022	Rp7.125	Rp7.206	Rp7.175	▲1,12%	▲0,70%	99,6%
11 Juli 2022	7	20 Juli 2022	Rp7.125	Rp7.149	Rp7.400	▲0,34%	▲3,72%	96,6%
12 Juli 2022	10	26 Juli 2022	Rp7.175	Rp7.135	Rp7.300	▼0,56%	▲1,71%	97,7%
12 Juli 2022	15	02 Agustus 2022	Rp7.175	Rp7.104	Rp7.600	▼1,00%	▲5,59%	93,5%
12 Juli 2022	20	09 Agustus 2022	Rp7.175	Rp6.996	Rp7.900	▼2,56%	▲9,18%	88,6%
							Rata-Rata	95,2%

TABEL 4.6

Hasil Prediksi Harga Saham MSFT Dalam Rentang Waktu Berbeda

TANGGAL DATA SAHAM TERAKHIR	PREDIKSI PADA HARI KE-X	TANGGAL YANG PREDIKSIKAN	HARGA CLOSING SEBELUM-NYA	PREDIKSI HARGA CLOSING	HARGA CLOSING REAL	PREDIKSI PERGERAKAN	PERGERAKAN REAL	NILAI AKURASI PREDIKSI
13 Juli 2022	1	14 Juli 2022	\$252,72	\$254,11	\$254,08	▲ 0,55%	▲ 0,54%	100,0%
13 Juli 2022	7	22 Juli 2022	\$252,72	\$262,24	\$262,27	▲ 3,63%	▲ 3,64%	100,0%
13 Juli 2022	10	27 Juli 2022	\$252,72	\$253,71	\$268,74	▲ 0,39%	▲ 5,96%	94,4%
13 Juli 2022	15	03 Agustus 2022	\$252,72	\$265,19	\$282,74	▲ 4,70%	▲ 10,62%	93,8%
13 Juli 2022	20	10 Agustus 2022	\$252,72	\$232,88	\$289,16	▼ 8,52%	▲ 12,60%	80,5%
							Rata-Rata	93,7%

Dapat dilihat pada tabel 4.5 dan 4.6 adalah hasil prediksi harga saham dengan *target* hari prediksi yang berbeda-beda, disini peneliti ingin melihat bagaimana hasil prediksi jika model prediksi digunakan untuk memprediksi beberapa hari kedepan, disini peneliti mencoba memprediksi pada hari ke 1, 7, 10, 15, dan 20. Dapat dilihat bahwa nilai akurasi dari masing-masing prediksi memiliki *trend* penurunan seiring bertambahnya hari yang diprediksi, peneliti berpendapat bahwa kemungkinan terbesar mengapa nilai akurasi mengalami penurunan karena jarak data dengan *target* prediksi yang terlalu jauh, mengingat bahwa harga saham sangat mudah berubah setiap harinya dikarenakan berbagai faktor, mulai dari faktor politik, sosial, ekonomi, dan faktor-faktor internal pada perusahaan. Setelah mendapatkan seluruh akurasi hasil prediksi tersebut, kemudian dilakukan perhitungan rata-rata keseluruhan yang didapatkan dengan perhitungan sebagai berikut :

$$R_t = \frac{\bar{x}}{n} \times 100\% \quad (11)$$

$$R_t = \frac{30.2}{31} \times 100\%$$

$$R_t = 0.974 \times 100\%$$

$$R_t = 97,4\%$$

Jadi, rata-rata keseluruhan akurasi prediksi adalah 97,4%

4.2.3. Hasil Pada *Client-Side*

The screenshot shows a web application titled "PREDIKSI HARGA SAHAM MENGGUNAKAN DEEP LEARNING". At the top, there is a header bar with the Anvil logo and the text "Build web apps for free with Anvil". Below the header, there is a form field labeled "Kode Saham:" with a placeholder "Masukkan Kode Saham". Below the form field, there are three small explanatory notes: "Untuk saat ini, hasil prediksi yang memiliki akurasi tinggi adalah saham MSFT(Microsoft) dan BBCA.JK (Bank Central Asia)", "Namun anda bisa menggunakan berbagai macam saham untuk melihat informasi saham", and "Untuk saham pada pasar saham IHSG, tambahkan ".JK" pada akhir kode saham". Below the form field are two blue buttons: "TAMPILKAN HASIL PREDIKSI" and "TAMPILKAN INFORMASI SAHAM". At the bottom of the screen, there are two sections: "Hasil Prediksi:" containing the text "Kode Saham : BBCA.JK", "Epochs : 1000", and "Prediksi Harga Closing : Rp/\$ 7907.83"; and "Informasi Saham:" containing the text "Saham : BBCA.JK", "Harga Penutupan Sebelumnya : 7950.0", "Permitaan : 7950.00 x 0", "Penawaran : 7925.00 x 0", "Volume Perdagangan : 61083100.0", "Kapitalisasi Pasar : 976.954T", and "Ratio P/E : 27.9".

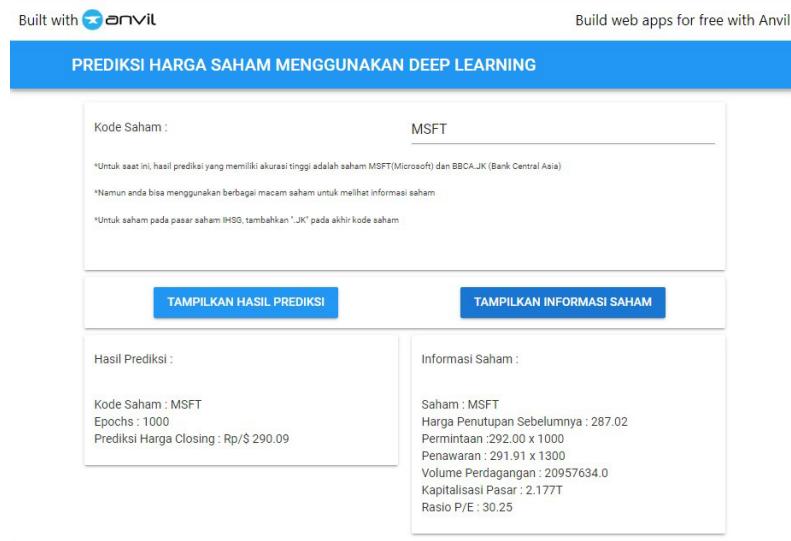
Gambar 4.8

Screenshot dari *client-side* UI aplikasi prediksi harga saham

The screenshot shows the same web application as in Figure 4.8, but with the "Hasil Prediksi:" section populated with the prediction results for BBCA stock. The "Hasil Prediksi:" section now displays "Kode Saham : BBCA.JK", "Epochs : 1000", and "Prediksi Harga Closing : Rp/\$ 7907.83". The "Informasi Saham:" section also displays more detailed information about BBCA stock, including its current price, trading volume, and market capitalization.

Gambar 4.9

Screenshot hasil prediksi saham BBCA pada *client-side*



Gambar 4.10

Screenshot hasil prediksi saham MSFT pada *client-side*

Dapat dilihat pada gambar 4.8 sampai 4.10 adalah *screenshot* dari UI *client-side* pada aplikasi prediksi harga saham, terlihat bahwa selain bisa untuk melihat prediksi, disini peneliti juga menambahkan fitur diluar *scope* penelitian ini, yaitu fitur untuk melihat informasi saham, informasi saham yang dapat dilihat pengguna berupa harga penutupan sebelumnya, permintaan dan penawaran saat saham diperdagangkan, volume perdagangan terakhir, kapitalisasi pasar perusahaan, dan rasio P/E atau rasio *price to earning*.

4.3. Implikasi Penelitian

Implikasi yang didapat dari penelitian dan pengembangan rancang bangun aplikasi prediksi harga saham menggunakan *deep learning* dengan algoritma *long short-term memory* adalah sebagai berikut :

a. Aspek Aplikasi

Penelitian pengembangan aplikasi prediksi harga saham menggunakan *deep learning* dengan algoritma *long short-term memory* dapat membantu memprediksi harga *closing* suatu saham, penelitian ini mampu menghasilkan prediksi dengan rata-rata keseluruhan akurasi

mencapai 97,4% yang didapatkan dengan menghitung nilai rata-rata dari seluruh hasil akurasi prediksi yang didapatkan.

Penggunaan aplikasi prediksi harga saham ini dapat membantu dan menjadi referensi bagi para investor baru maupun investor berpengalaman untuk mengetahui prediksi harga *closing* suatu saham. Penggunaan aplikasi prediksi harga saham ini dapat membantu para investor baru untuk memilih saham perusahaan mana yang akan dibeli dengan melihat prediksi dan informasi saham yang tersedia.

b. Aspek Penelitian Lanjutan

Diperlukannya pelatihan atau *training* model prediksi lebih banyak dengan menggunakan berbagai macam saham agar pengguna bisa mendapatkan hasil prediksi dari lebih banyak saham, diperlukan juga pelatihan model prediksi menggunakan *hardware* dengan tingkat komputasi yang tinggi untuk mengurangi waktu latihan, dengan waktu latihan berkurang maka bisa dilakukan lebih banyak latihan dalam satu waktu.

BAB V

KESIMPULAN DAN SARAN

1.1. Kesimpulan

Berdasarkan dari rumusan masalah yang telah dibuat, kesimpulan dari hasil penelitian dan pengembangan rancang bangun aplikasi prediksi harga saham menggunakan *deep learning* dengan algoritma *long short-term memory* yaitu :

- a. Cara membangun suatu aplikasi prediksi harga saham menggunakan *deep learning* dengan algoritma *long short-term memory* adalah sebagai berikut :
 - 1) Mengumpulkan data perdagangan saham yang ingin diprediksi menggunakan sebuah API yaitu API Yahoo Finance.
 - 2) Setelah data perdagangan saham dikumpulkan selanjutnya adalah data *preprocessing* yaitu sebuah proses yang terdiri dari 3 proses, ketiga proses tersebut adalah memasukkan data kedalam program, menskalakan data, dan membagi data menjadi dua *dataset*, yaitu *dataset training* dan *dataset latih*.
 - 3) Kemudian membangun model LSTM dengan menggunakan beberapa parameter, parameter yang digunakan adalah N_LAYERS, UNITS, CELL, DROPOUT, LOSS, OPTIMIZER, BIDIRECTIONAL, N_FEATURES, SEQUENCE_LENGTH.
 - 4) Berikutnya adalah melatih model LSTM menggunakan kedua *dataset training* dan *dataset latih* dengan parameter EPOCHS dan BATCH_SIZE. Setelah model dilatih maka versi terbaik dari model hasil latihan disimpan.
 - 5) Berikutnya adalah proses prediksi, model terbaik yang telah dilatih kemudian dipanggil dan digunakan untuk proses prediksi, proses ini menggunakan beberapa parameter diantaranya N_STEPS, LOOKUP_STEP, LOSS, CELL,

N_LAYERS, OPTIMIZER, DROPOUT, BIDIRECTIONAL, FEATURE_COLUMNS.

- 6) Setelah proses prediksi selesai maka hasil prediksi ditampilkan, hasil prediksi kemudian dihitung tingkat akurasi dengan harga saham sebenarnya menggunakan rumus akurasi prediksi yang telah dijelaskan pada bab 2 penelitian ini. Untuk penelitian ini, didapatkan rata-rata keseluruhan akurasi sebesar 97,4% yang dihitung dari nilai rata-rata seluruh hasil akurasi prediksi mulai dari tabel 4.3 hingga tabel 4.6.

1.2. Saran

Saran yang dapat disampaikan untuk aplikasi prediksi harga saham menggunakan *deep learning* dengan algoritma *long short-term memory* untuk kedepannya adalah :

- a. Diperlukannya *training* dengan *hardware* spesifikasi tinggi, *hardware* yang memiliki tingkat komputasi cepat untuk menyingkat waktu latihan model prediksi.
- b. Dengan lebih cepatnya waktu latihan model prediksi, maka dapat dilakukan lebih banyak latihan model prediksi untuk berbagai saham lain yang akan memudahkan para pengguna mengetahui hasil prediksi dari lebih banyak saham.
- c. Dengan lebih cepatnya waktu latihan model prediksi, maka dapat dilakukan lebih banyak percobaan untuk meningkatkan akurasi dari prediksi yaitu dengan meningkatkan jumlah *epochs* dan jumlah *batch_size*.
- d. Dengan menggunakan *hardware* yang memiliki tingkat komputasi lebih cepat maka dapat dilakukan lebih banyak *tuning* dengan cara merubah beberapa parameter latihan dan prediksi untuk melihat apakah hasil prediksi lebih akurat.

DAFTAR PUSTAKA

- Akita, R., Yoshihara, A., Matsubara, T., & Uehara, K. (2016). Deep learning for stock prediction using numerical and textual information. *2016 IEEE/ACIS 15th International Conference on Computer and Information Science, ICIS 2016 - Proceedings*. <https://doi.org/10.1109/ICIS.2016.7550882>
- Aprian, B. A. (2020). *PREDIKSI PENDAPATAN CARGO SERVICE CENTER TANGERANG CITY MENGGUNAKAN ARSITEKTUR LONG SHORT TERM MEMORY*. Universitas Muhammadiyah Malang.
<https://eprints.umm.ac.id/62460/>
- Awal, S. (2022). *Cara Membaca Candlestick*. Stockbit.
<https://snips.stockbit.com/investasi/cara-membaca-candlestick>
- BD TechTalk. (2019). *Artificial Neural Network Structure*.
<https://bdtechtalks.com/2019/08/05/what-is-artificial-neural-network-ann/artificial-neuron/>
- Bursa Efek Indonesia (BEI). (2010). *Definisi Saham*.
- Colah. (2015). *Understanding LSTM*. <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- Dhea Larasati, K. (2019). *Artificial Neural Network Image*.
<https://medium.com/@dhea.larasati326/artificial-neural-network-55797915f14a>
- Dolphin, R. (2020). *Long Short-Term Memory*.
<https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9>
- EdrawMax. (2020). *Use-Case Diagram Symbol*.
<https://www.edrawmax.com/article/use-case-diagram-uml.html>

- Google. (2017). *Google Colab*. <https://research.google.com/colaboratory/faq.html>
- IBM. (2020a). *Deep Learning*. [https://www.ibm.com/cloud/learn/deep-learning#:~:text=Deep learning attempts to mimic,make predictions with incredible accuracy.](https://www.ibm.com/cloud/learn/deep-learning#:~:text=Deep%20learning%20attempts%20to%20mimic,make%20predictions%20with%20incredible%20accuracy.)
- IBM. (2020b). *Recurrent Neural Network*.
<https://www.ibm.com/cloud/learn/recurrent-neural-networks>
- Lucid Chart. (2019). *Use-Case Diagram*. <https://www.lucidchart.com/pages/uml-use-case-diagram>
- McCarthy, J. (1956). *Definition Of Artificial Intelligence*. <http://www-formal.stanford.edu/jmc/whatisai/node1.html>
- Mubarok, H. (2019). Identifikasi Ekspresi Wajah Berbasis Citra Menggunakan Algoritma Convolutional Neural Network (CNN). In *Universitas Islam Negeri Maulana Malik Ibrahim Malang* (Vol. 3, Nomor 1).
- Phi, M. (2018). *LSTM Process*. Towards DataScience.
<https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>
- Rastogi, M. (2020). *LSTM Tutorial*. <https://towardsdatascience.com/tutorial-on-lstm-a-computational-perspective-f3417442c2cd>
- Rosebrock, A. (2017). Deep Learning for Computer Vision with Python - Starter Bundle. *Pyimagesearch*.
- Sagar Sharma. (2017). *Activation Function*.
<https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>
- Smart Draw. (2019). *Activity Diagram*. [https://www.smartdraw.com/activity-diagram/#:~:text=Activity Diagram-,What is an Activity Diagram%3F,in a](https://www.smartdraw.com/activity-diagram/#:~:text=Activity%20Diagram-,What%20is%20an%20Activity%20Diagram%3F,in%20a)

use case diagram.

Tensorflow, & Google. (2012). *Tensorflow*. <https://www.tensorflow.org/about>

Yahoo Finance. (2022). *Stock Data*. <https://finance.yahoo.com/>

Yasin, H., Prahutama, A., & Utami, T. W. (2014). PREDIKSI HARGA SAHAM MENGGUNAKAN SUPPORT VECTOR REGRESSION DENGAN ALGORITMA GRID SEARCH. *MEDIA STATISTIKA*, 7(1).
<https://doi.org/10.14710/medstat.7.1.29-35>

Zafeirios Fountas. (2011). *Spiking Neural Networks for Human-like Avatar Control in a Simulated Environment*.