

RECURRENT NEURAL NETWORK DENGAN GATE RECURRENT UNIT UNTUK PREDIKSI HARGA SAHAM

TUGAS AKHIR



DISUSUN OLEH :

AFIF ILHAM CANIAGO

123160063

PROGRAM STUDI INFORMATIKA

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK INDUSTRI

UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN" YOGYAKARTA

2021

HALAMAN PENGESAHAN PEMBIMBING

RECURRENT NEURAL NETWORK DENGAN GATE RECURRENT UNIT UNTUK PREDIKSI HARGA SAHAM

Disusun oleh:
Afif Ilham Caniago
123160063

Telah diperiksa dan disetujui oleh pembimbing untuk diseminarkan pada
tanggal: 26 Februari 2021


Menyetujui,
Pembimbing I

Pembimbing II


Wilis Kaswidjanti, S.Si., M.Kom
NIK. 2 7604 00 0226 1


Juwairiah, S.Si., M.T.
NIK. 2 7607 00 0230 1

Mengetahui, Koordinator Tugas Akhir


Dr. Awang Hendrianto Pratomo, S.T., M.T.
NIP. 1977 07 25 2005 01 1001

DAFTAR ISI

JUDUL	i
LEMBAR PENGESAHAN.....	ii
DAFTAR ISI	iii
DAFTAR GAMBAR.....	v
DAFTAR TABEL	vi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah.....	2
1.3 Batasan Masalah	2
1.4 Tujuan Penelitian	3
1.5 Manfaat Penelitian	3
1.6 Metodologi Penelitian.....	3
1.6.1 Metodologi Pengumpulan Data	3
1.6.2 Metodologi Penelitian	3
1.7 Sistematika Penulisan	4
BAB II TINJAUAN LITERATUR	5
2.1 Investasi	5
2.2 Pengertian Saham	5
2.3 Pasar Modal	5
2.4 Analisis Saham	6
2.5 <i>Data Mining</i>	9
2.5.1 Min Max Scaler	11
2.5.2 <i>Sliding windows</i>	11
2.5.3 <i>Mean Absolute Error</i> (MAE)	12
2.5.4 <i>Root Mean Squared Error</i> (RMSE)	12
2.5.5 <i>Directional Accuracy</i> (DA)	13
2.6 Deep Learning.....	13
2.7 Artificial Neural Network (ANN).....	13
2.8 <i>Gradient Descent</i>	17
2.9 Recurrent Neural Networks (RNN)	21
2.10 Gated Recurrent Unit (GRU)	23

2.11	Parameter dan Arsitektur	25
2.12	<i>Dropout</i>	25
2.13	Penelitian Terdahulu	26
BAB III Metodologi Penelitian dan Pengembangan Sistem		6
3.1	Metodologi Penelitian	6
3.1.1	Pengumpulan Data	30
3.1.2	<i>Praprocessing</i> data	32
3.1.3	Proses Pra pengujian arsitektur	40
3.1.4	Pengujian Arsitektur	41
3.1.5	Pengujian <i>Dropout</i>	52
DAFTAR PUSTAKA		55

DAFTAR GAMBAR

Gambar 2.1 Ilustrasi <i>Sliding windows</i>	12
Gambar 2.2 Model ANN	14
Gambar 2.3 Model <i>neuron</i>	14
Gambar 2.4 Fungsi <i>Sigmoid</i>	15
Gambar 2.5 Fungsi TanH	16
Gambar 2.6 Fungsi ReLU.....	16
Gambar 2.7 Fungsi Leaky ReLU.....	17
Gambar 2.8 <i>Backpropagation</i>	18
Gambar 2.9 Ilustrasi <i>batch</i>	20
Gambar 2.10 Arsitektur RNN.....	21
Gambar 2.11 Detail kerja sel RNN.....	22
Gambar 2.12 Arsitektur Modul GRU	24
Gambar 3.1 Metodologi Penelitian.....	30
Gambar 3.2 Data IDX per hari	30
Gambar 3.3 Pergerakan harga penutupan ADRO	31
Gambar 3.4 <i>Praprocessing</i> TA.....	32
Gambar 3.5 Ilustrasi Selisih Harga Penutupan.....	36
Gambar 3.6 Ilustrasi <i>Sliding Window</i>	38
Gambar 3.7 <i>Flowchart Sliding Window</i>	39
Gambar 3.8 Pelabelan.....	39
Gambar 3.9 <i>Flowchart Min Max Scaler</i>	40
Gambar 3.10 Ilustrasi arsitektur RNN-GRU	42
Gambar 3.11 <i>Flowchart Training</i> RNN-GRU.....	43
Gambar 3.12 <i>Flowchart Feed forward</i> RNN-GRU.....	44
Gambar 3.13 <i>Backpropagation Through Time</i>	48
Gambar 3.14 <i>Flowchart Backpropagation Through Time</i>	48
Gambar 3.15 <i>Flowchart</i> ADAM	49
Gambar 3.16 Alur Pengujian satu perusahaan.....	50
Gambar 3.17 Ilustrasi layer <i>Dropout</i>	53

DAFTAR TABEL

Tabel 2.1 Tabel Penelitian Sebelumnya	26
Tabel 2.2 Tabel Penelitian Sebelumnya(lanjutan).....	27
Tabel 2.3 Tabel Penelitian Sebelumnya(lanjutan).....	28
Tabel 3.1 Dataset 31 Desember 2019	30
Tabel 3.2 Dataset Adaro Energy (ADRO).....	31
Tabel 3.3 Daftar Dataset yang digunakan	32
Tabel 3.4 Parameter <i>Technical Analysis</i>	32
Tabel 3.5 Ilustrasi perhitungan MACD	36
Tabel 3.6 Contoh dataset	39
Tabel 3.7 Contoh <i>Sliding Window</i>	39
Tabel 3.9 Parameter RNN-GRU tetap	41
Tabel 3.10 Contoh dataset TA	43
Tabel 3.11 Contoh inisialisasi parameter RNN-GRU	43
Tabel 3.13 Pengujian <i>Shallow</i>	50
Tabel 3.14 Pengujian <i>Stacked</i>	50
Tabel 3.15 Pengujian <i>Stacked</i> (lanjutan)	51
Tabel 3.16 Pengujian <i>Stacked</i> (lanjutan)	52
Tabel 3.16 Pengujian Regulasi <i>Dropout</i>	52

BAB I PENDAHULUAN

1.1 Latar Belakang

Pasar saham atau bursa efek adalah wahana dimana dilakukan perdagangan saham dan instrumen finansial lainnya. Pasar saham berfungsi memfasilitasi antara pihak yang membutuhkan pendanaan (perusahaan) dan pihak yang mempunyai dana (Wira, 2014). Saham merupakan surat berharga yang memberikan peluang keuntungan yang tinggi namun juga berpotensi risiko tinggi. Dengan berfluktuasinya harga saham, saham juga dapat membuat investor mengalami kerugian besar dalam waktu singkat. Fluktuasi ini terjadi karena faktor-faktor ekonomi yang terjadi secara tiba-tiba dan susah diprediksi. Untuk itu investor membutuhkan sistem prediksi yang dapat membantunya dalam mengambil keputusan investasi pembelian saham.

Data pada pergerakan harga saham memiliki tiga isu yaitu bersifat *nonlinear*, *nonstationary*, dan memiliki *long memory dependency* (Faurina, 2019). Pergerakan harga saham dipengaruhi berbagai hal namun sebagian besar faktor ekonomi, performa perusahaan dan psikologi pasar adalah faktor yang sangat berpengaruh. Hal ini membuat pergerakan saham bersifat *nonlinear* dan *nonstationary* (Faurina, 2019). Secara statistik para pakar saham menggunakan analisis teknikal dalam memprediksi harga saham. Namun teknik ini bersifat subjektif dan bias analisis karena analisis pakar satu dengan yang lainnya dapat berbeda tergantung dari pengalaman dan pemilihan *Technical Indicator* yang digunakan (Faurina, 2019).

Data pergerakan harga saham merupakan data *time series* Sehingga membuat adanya keterikatan pengaruh data masa lampau secara beruntun. Pendekatan komputasional telah banyak dilakukan dalam memprediksi pergerakan saham. Ratnayaka dkk (2015) membandingkan metode *Autoregressive integrated moving average* (ARIMA), *Artificial Neural Network* (ANN) dan *hybrid ANN-ARIMA* dengan dua dataset berbeda yaitu ASPI dan SL20 dengan hasil pada dataset ASPI adalah MAE ARIMA sebesar 62.7512, MAE ANN sebesar 36.9834, MAE ANN-ARIMA sebesar 21.2208 dan dataset SL20 adalah MAE ARIMA sebesar 28.8660, MAE ANN sebesar 19.8767, MAE ANN-ARIMA sebesar 14.1000. Pada penelitian Ratnayaka dkk (2015) nilai error pada ANN-ARIMA dan ANN relatif rendah (Ratnayaka, Seneviratne, Jianguo, & Arumawadu, 2015). Yakup dkk (2011) membandingkan *neural network* dan *Support Vector Machine* (SVM). Rata-rata performa akurasi yang menggunakan metode T-test adalah *Neural network* (75,74%) lebih baik ketimbang SVM (71,52%) (Kara, Acar Boyacioglu, & Baykan, 2011). Erkam dkk (2011) melakukan penelitian terhadap beberapa metode *neural network* yaitu *multi-layer perceptron* (MLP), *Dynamic artificial neural network* (DAN2) dan *hybrid neural networks* yang menggunakan *generalized autoregressive conditional heteroscedasticity* (GARCH) dengan hasil uji MLP mempunyai MSE sebesar 2478,1468, MAD sebesar 41,153, MAD% sebesar 2,516. Hasil GARCH-MLP mempunyai MSE sebesar 3665,8387, MAD sebesar 42,739, MAD% sebesar 2,775. Hasil DAN2 mempunyai MSE sebesar 1472,278, MAD sebesar 32,875, MAD% sebesar 2,768 (Guresen, Kayakutlu, & Daim, 2011). *Neural network* merupakan metode yang sering dipakai ketika menghadapi data yang bersifat

nonlinear dan *nonstationary*. Ma dkk (2020) mengatakan ANN (*Artificial Neural Network*) dapat digunakan untuk memprediksi ANN dapat memprediksi data dimensi data yang tinggi, dapat mempelajari fluktuasi dari data yang *nonlinear* dan *nonstationary* (Ma, Antoniou, & Toledo, 2020)

Salah satu model dari ANN yang cukup populer dalam prediksi dan klasifikasi data *time series* adalah *Recurrent Neural Networks* (RNN). RNN cocok dipakai pada dataset yang bersifat *time series* karena model ini memiliki kemampuan dalam mempelajari data *sequence* di mana output jaringan bergantung pada jumlah yang berubah ubah dari input sebelumnya. RNN dapat menjadi model yang berguna untuk membuat prediksi deret waktu yang tidak beraturan, terutama dalam membuat prediksi *multi-step* (Id, Abbas, & Turel, 2019). Namun RNN memiliki kelemahan dalam menghadapi prediksi dengan *frame* waktu yang cukup panjang yang mana permasalahan ini lebih dikenal dengan *long memory dependency*. Karena RNN akan mengalami permasalahan yang dinamakan *vanishing gradient* dan *gradient exploding* apabila RNN menangani masalah prediksi dengan panjang *range* waktu yang panjang (Wang, Yan, Li, Gao, & Zhao, 2019). *Gated Recurrent Unit* (GRU) datang sebagai pengembangan lebih lanjut dari unit RNN konvensional yang memiliki kemampuan dalam menangani kasus prediksi dan klasifikasi tanpa adanya permasalahan *vanishing gradient* dan *gradient exploding* (Wang et al., 2019). GRU merupakan unit RNN yang merupakan sebuah model *machine learning* merupakan pengembangan dari unit *Long Short Term Memory* (LSTM). GRU merupakan penyederhanaan dari LSTM yang berupa pengurangan *gate* dan parameter untuk mempercepat waktu pelatihan dan mempermudah implementasi (Althelaya, El-Alfy, & Mohammed, 2018).

Dari permasalahan tersebut maka dapat dirancang sebuah model *machine learning* dengan menggunakan metode *Recurrent Neural Networks* dengan unit *Gated Recurrent Unit* RNN-GRU dengan melakukan evaluasi terhadap beberapa parameter baik dari sisi data maupun dari sisi model dengan harapan model GRU dapat melakukan prediksi terhadap pergerakan harga saham dengan akurasi yang tinggi.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang dikemukakan, maka dapat dirumuskan masalah yaitu Evaluasi parameter pada model RNN-GRU yang mempengaruhi tingkat akurasi dalam memprediksi harga saham yang merupakan data *time series* dengan sifat data *nonlinear*, *nonstationary*, dan *long memory dependency*.

1.3 Batasan Masalah

Agar masalah yang dibahas menjadi fokus dan lebih jelas dalam mencapai sasaran, maka dibuat batasan dari perumusan masalah di atas, diantaranya adalah:

1. Sumber data dari penelitian ini adalah data ringkasan saham yang terdapat pada situs web Bursa Efek Indonesia (idx.co.id).
2. Data perusahaan yang digunakan adalah data perusahaan yang mempunyai *historical* data yang lengkap dari tahun Januari 2015 – Desember 2019.
3. Prediksi yang dilakukan berdasarkan statistik dari data pergerakan harga saham pada masa lalu. Prediksi tidak berdasarkan faktor fundamental seperti performa perusahaan, situasi kepanikan, situasi politik, dan keadaan ekonomi.

4. Prediksi harga saham jangka pendek dengan rentang waktu prediksi yang digunakan adalah satu hari, tiga hari, satu minggu, dua minggu, dan satu bulan.
5. Saham yang diprediksi merupakan jenis saham biasa (*Common Stock*)
6. Data saham yang digunakan adalah PT. Adaro Energy Tbk, PT. Vale Indonesia Tbk, PT. Astra International Tbk, PT. Indomobil Sukses Internasional Tbk, PT. Telekomunikasi Indonesia (Persero) Tbk, dan PT. Garuda Indonesia (Persero) Tbk.

1.4 Tujuan Penelitian

Tujuan dari pembuatan tugas akhir ini untuk melakukan prediksi terhadap harga saham menggunakan metode *Recurrent Neural Networks* dengan *Gated Recurrent Unit* dan mengevaluasi parameter yang mempengaruhi kinerja model RNN-GRU dalam melakukan prediksi harga saham.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini agar dapat membantu para broker dan investor saham agar dapat meminimalisasi kerugian dalam berbisnis saham. Dengan adanya prediksi ini diharapkan para broker dan investor dapat mengambil keputusan dalam jual beli saham agar kerugian yang didapat dapat dikurangi. Dengan hasil prediksi ini para pelaku bisnis saham juga dapat memaksimalkan keuntungan yang diperoleh.

1.6 Metodologi Penelitian

Metode penelitian yang digunakan dalam penelitian ini adalah sebagai berikut:

1.6.1 Metodologi Pengumpulan Data

1. Studi Literatur dan Analisis Kebutuhan
2. Studi literatur dilakukan dengan cara membaca jurnal, skripsi, maupun tesis yang mendukung penelitian untuk menyelesaikan permasalahan pada penelitian ini.
3. Validasi Data
4. Data yang akan digunakan bersumber dari web Bursa Efek Indonesia yaitu pada bagian Ringkasan saham.
5. Data yang dikumpulkan akan diolah terlebih dahulu agar dapat terbaca dan terstruktur.

1.6.2 Metodologi Pengujian

1. Data *praprocessing*
Pada tahap ini data akan ditransformasi agar dapat diterima oleh model RNN-GRU dan dapat menghasilkan model yang lebih baik daripada tanpa *praprocessing*. Metode *praprocessing* yang dilakukan sesuai dengan permasalahan.
2. Pengujian Arsitektur
Proses ini adalah proses pelatihan yang akan menguji parameter model RNN-GRU berupa jumlah unit, jenis arsitektur *hidden layer*, dan jenis data input(*technical analysis* dan *chartist*). Ketiga parameter itu akan dikombinasikan sehingga mendapatkan model yang terbaik. Pengujian arsitektur akan diukur dari seberapa

tepat dan akurat model memprediksi saham pada data *training* dan data validasi dengan ukuran *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), dan *Directional Accuracy* (DA).

3. Pengujian Regulasi *Dropout*

Proses ini berguna dalam meminimalisir indikasi *overfitting* pada model RNN-GRU. Apabila model *overfitting* maka pengujian akan menjadi bias karena hanya dapat memprediksi data *training* di penelitian ini namun tidak bisa memprediksi data baru.

1.7 Sistematika Penulisan

Rencana penyusunan Tugas Akhir akan dilakukan secara sistematis. Adapun sistematika penulisan laporan ini sebagai berikut :

Bab I Pendahuluan

Pada bagian ini membahas tentang latar belakang masalah, perumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan sistematika penulisan.

Bab II Tinjauan Literatur

Tinjauan pustaka memuat tentang dasar teori yang digunakan untuk analisis dan perancangan aplikasi serta implementasi pada penelitian ini. Selain itu juga sebagai bahan referensi dan pondasi untuk memperkuat argumen dalam penelitian ini. Teori-teori yang sesuai dengan penelitian ini antara lain pasar saham, analisis teknikal pergerakan harga saham, teknikal indikator harga saham dan GRU.

Bab III Metodologi Penelitian dan Pengembangan Sistem

Pada bagian ini akan membahas mengenai analisis dan perancangan dan desain hingga implementasi program.

Bab IV Hasil, Pengujian dan Pembahasan

Pada BAB empat akan menyajikan hasil penelitian yang berisi hasil implementasi dari perancangan yang telah dibuat pada BAB sebelumnya. Selain itu berisi analisis terhadap hasil penelitian.

Bab V Kesimpulan dan Saran

Pada bagian ini berisi kesimpulan dari hasil penelitian dan saran yang diajukan oleh penulis untuk pengembangan penelitian selanjutnya.

BAB II TINJAUAN LITERATUR

2.1 Investasi

Investasi merupakan suatu usaha penanaman modal pada suatu instrumen yang nyata maupun yang tidak nyata dengan harapan pada waktunya nanti investor akan mendapatkan keuntungan. Dalam hal ini investasi mempunyai tujuan untuk memanfaatkan aset yang dimiliki seseorang untuk hal yang lebih bernilai agar dapat menambah dan meningkatkan kesejahteraan pemilik aset (Arista, 2012). Investasi merupakan variabel ekonomi yang merupakan penghubung antara kondisi saat ini dengan masa yang akan datang, serta menghubungkan antara pasar barang dengan pasar uang (Rasyidin, 2014).

Seseorang yang berinvestasi dalam suatu barang atau perusahaan mengharapkan suatu keuntungan dari usaha penanaman modal tersebut. Investor dapat mengalami kerugian dalam berinvestasi karena nilai uang yang kembali kepada investor lebih kecil dari pada jumlah modal yang dikeluarkan. Oleh sebab itu investor harus hati-hati dalam berinvestasi pada sebuah instrumen tertentu.

2.2 Pengertian Saham

Saham merupakan bukti penyertaan modal pada sebuah perusahaan. Individu yang membeli saham perusahaan artinya berinvestasi pada sebuah perusahaan yang nantinya modal tersebut akan digunakan oleh pihak manajemen untuk membiayai kegiatan operasional perusahaan. Timbal balik dari manajemen perusahaan terhadap para investor adalah berupa dividen (Tambunan, 2007). Dividen merupakan pembayaran dari perusahaan kepada para pemegang saham atas keuntungan laba bersih yang diperolehnya (Sutrisno, 2001). Ada dua jenis saham berdasarkan sistem pembayaran dividen (Tambunan, 2007) yaitu:

1. Saham Preferen (*Preferred Stock*), yaitu penanaman modal atau kepemilikan pada suatu perusahaan pada tingkat batas. Investor yang mempunyai kepemilikan pada saham ini tidak memiliki hak suara dalam Rapat Umum Pemegang Saham (RUPS). Namun demikian, investor akan dijanjikan sejumlah dividen yang jumlahnya tetap. Investor akan memperoleh dividen tetap walaupun laba bersih di suatu perusahaan mengalami kenaikan atau penurunan. Saham ini adalah jenis saham yang dinilai paling aman dari krisis perusahaan. Namun memiliki kekurangan apabila saham perusahaan mengalami kenaikan laba bersih. Saham ini akan dibagikan terlebih dahulu daripada saham biasa.
2. Saham Biasa (*Common Stock*), yaitu penanaman modal atau kepemilikan pada suatu perusahaan melalui penyertaan modal (*Equity Investment*). Investor yang mempunyai kepemilikan pada saham ini hak suara dalam Rapat Umum Pemegang Saham (RUPS). Saham jenis ini adalah investasi yang berisiko namun juga sangat menguntungkan apabila perusahaan mengalami kenaikan performa perusahaan.

2.3 Pasar Modal

Dunia bisnis sekarang ini mengalami perkembangan yang sangat pesat, dapat dilihat dari banyak perusahaan-perusahaan baru yang bermunculan dengan keunggulan kompetitif.

Dalam mengembangkan usahanya perusahaan memerlukan tambahan modal yang bisa didapatkan melalui berbagai cara, salah satunya dengan memutuskan untuk *Go Public* (Dewi & Suaryana, 2013). Perusahaan yang sudah *Go Public* menawarkan sahamnya kepada publik untuk meraih dana dari masyarakat penanam modal (investor). Tujuan dari perusahaan *Go Public* adalah untuk ekspansi usaha, mengakuisisi perusahaan lain, membayar utang, dan lain sebagainya (Tambunan, 2007).

Media bagi perusahaan *Go Public* adalah pasar modal. Pasar modal adalah Media untuk mempertemukan antara perusahaan dan investor. Pasar modal atau bursa efek adalah pasar untuk perdagangan saham perusahaan yang dipegang umum dan instrumen finansial yang berhubungan (termasuk opsi saham, perdagangan dan prakiraan indeks saham). Bursa efek atau pasar saham merupakan sarana pembiayaan melalui penerbitan saham, dan merupakan sarana perdagangan saham. Bursa efek adalah tempat dimana perusahaan dapat menawarkan sahamnya untuk dijual.

Bursa Efek yang ada di Indonesia dikenal dengan nama Bursa Efek Indonesia. Sebelumnya ada 2 bursa efek, yaitu Bursa Efek Jakarta dan Bursa Efek Surabaya. Namun, sekarang semua kegiatan transaksi jual beli modal terpusat semuanya di Jakarta. Bursa efek merupakan tempat untuk melakukan aktivitas jual beli segala macam instrumen pasar modal, dengan 2 peranan besar (Ratri, 2020), yaitu:

1. Fasilitator perdagangan, dengan tugas Menyediakan semua fasilitas yang diperlukan untuk melaksanakan kegiatan jual beli saham dengan lancar, Membuat peraturan yang harus dipatuhi oleh para pelaku, Mencatat semua aktivitas yang terjadi di dalam perdagangan efek, Bertanggung jawab atas likuiditas semua instrumen yang diperdagangkan, Memberikan informasi sejelas-jelasnya mengenai kegiatan transaksi yang terjadi.
2. Kontrol perdagangan efek yang terjadi, dengan wewenang Memantau setiap kegiatan yang terjadi di dalam bursa, Mencegah terjadinya kejahatan di pasar saham, Membekukan atau suspending terhadap peserta bursa yang melanggar peraturan, Melakukan delisting sesuai peraturan yang berlaku.

2.4 Analisis Saham

Saham merupakan investasi yang mempunyai risiko. Apabila performa dari perusahaan bagus maka investor akan menerima keuntungan (*Capital Gain*) namun apabila perusahaan tidak dalam keadaan yang buruk maka investor akan mendapatkan kerugian (*Capital Loss*). Untuk menghindari *capital loss* maka para investor dapat menganalisis saham yang akan dibeli. Analisis Saham merupakan suatu tahapan untuk memprediksi pergerakan dan trend dari suatu perdagangan saham Pendekatan analisis yang ditawarkan oleh para analis dalam menganalisis pergerakan nilai saham dibagi menjadi dua yaitu analisis fundamental dan teknikal (Hendarsih, 2008).

1. Analisis Fundamental

Analisis fundamental merupakan faktor yang erat kaitannya dengan kondisi perusahaan yaitu kondisi manajemen organisasi sumber daya manusia dan kondisi keuangan perusahaan yang tercermin dalam kinerja keuangan perusahaan. Analisis fundamental merupakan pendekatan analisis harga saham yang menitikberatkan pada

kinerja perusahaan yang mengeluarkan saham dan analisis ekonomi yang akan mempengaruhi masa depan perusahaan (Sutrisno, 2005). Sebagian pakar berpendapat teknik analisis fundamental lebih cocok untuk membuat keputusan dalam memilih saham perusahaan mana yang dibeli untuk jangka panjang. Beberapa faktor utama atau fundamental yang mempengaruhi harga saham yaitu penjualan, pertumbuhan penjualan, operasional perusahaan, laba, dividen, Rapat Umum Pemegang Saham (RUPS), perubahan manajemen, dan pernyataan-pernyataan yang dibuat oleh manajemen perusahaan.

2. Analisis Teknikal

Menurut (Sutrisno, 2005) analisis teknikal adalah pendekatan investasi dengan cara mempelajari data historis dari harga saham serta menghubungkannya dengan trading volume yang terjadi dan kondisi ekonomi pada saat itu. Analisis teknikal menurut (David, 2010) adalah suatu jenis analisis yang selalu berorientasi kepada harga (pembukaan, penutupan, tertinggi dan terendah) dari suatu instrumen investasi pada batas waktu tertentu (berorientasi terhadap harga). Analisis teknikal merupakan upaya untuk memperkirakan harga saham dengan mengamati perubahan harga saham di periode yang lalu dan upaya untuk menentukan kapan investor harus membeli, menjual atau mempertahankan sahamnya dengan menggunakan indikator-indikator teknis atau menggunakan analisis grafik.

Analisis teknis dapat dilakukan dengan analisis grafik dan menggunakan indikator-indikator teknis. Indikator teknis (*Teknikal Indicator*) berguna dalam memprediksi kecenderungan arah pergerakan saham. Banyak indikator saham yang digunakan dalam memprediksi kecenderungan pergerakan harga saham. Berikut adalah beberapa *technical indicator* (Kara et al., 2011) :

1. Simple 10-day Moving Average (SMA10)

$$SMA10 = \frac{C_t + C_{t-1} + \dots + C_{t-9}}{10} \dots\dots\dots (2.1)$$

Keterangan :

C_t = Closing price pada waktu ke t

2. Weighted 10-day Moving Average (WMA10)

$$WMA10 = \frac{(n) \times C_t + (n-1) \times C_{t-1} + \dots + (n-9) \times C_{t-9}}{(n) + (n-1) + (n-2) + \dots + 1} \dots\dots\dots (2.2)$$

Keterangan :

n = Bobot (karena rentang waktunya 10 hari maka n bernilai 10)

C_t = Closing price pada waktu ke t

3. Momentum (M)

$$M = C_t - C_{t-n} \dots\dots\dots (2.3)$$

Keterangan :

n = periode waktu

C_t = Closing price pada waktu ke t

4. *Stochastic K% (K%)*

$$K\% = \frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} \times 100 \dots\dots\dots (2.4)$$

Keterangan :

C_t = Closing price pada waktu ke t

LL_{t-n} = lowest low pada akhir hari ke-n dari hari ke-t

HH_{t-n} = highest high pada akhir hari ke-n dari hari ke-t

5. *Stochastic D% (D%)*

$$D\% = \frac{\sum_{i=0}^{n-1} K_{t-i}\%}{n} \dots\dots\dots (2.5)$$

Keterangan :

$K_{t-i}\%$ = *Stochastic K%* dari hari ke-t sampai hari ke-i

n = rentang waktu (standard waktu yang digunakan 3 hari)

6. *Relative Strength Index (RSI)*

$$RSI = 100 - \frac{100}{1 + \frac{(\sum_{i=0}^{n-1} Up_{t-i}/n)}{(\sum_{i=0}^{n-1} Dw_{t-i}/n)}} \dots\dots\dots (2.6)$$

Keterangan :

Up_t = perubahan harga ketika naik pada waktu ke-t

Dw_t = perubahan harga ketika turun pada waktu ke-t

7. *Moving Average Convergence Divergence (MACD)*

$$MACD = EMA(12)_t - EMA(26)_t \dots\dots\dots (2.7)$$

$$EMA(k)_t = EMA(k)_{t-1} + \alpha \times (C_t - EMA(k)_{t-1}) \dots\dots\dots (2.8)$$

$$\alpha = \frac{2}{1+k} \dots\dots\dots (2.9)$$

Keterangan :

EMA = *Exponential Moving Average*

8. *Larry William's R% (R%)*

$$R\% = \frac{H_n - C_t}{H_n - L_n} \times 100 \dots\dots\dots (2.10)$$

Keterangan :

H_n = high price pada waktu ke n

C_t = closing price pada waktu ke t

L_n = low price pada waktu ke n

9. Accumulation / Distribution Oscillator (A/D)

$$A/D = \frac{C_t - O_t}{(H_t - L_t) * V} \dots\dots\dots (2.11)$$

Keterangan :

O_t = Open price pada waktu ke t
 C_t = closing price pada waktu ke t
 H_t = high price pada waktu ke t
 L_t = low price pada waktu ke t
 V = Volume

10. Commodity Channel Index (CCI)

$$CCI = \frac{M_t - SM_t}{0.015 D_t} \dots\dots\dots (2.12)$$

$$M_t = \frac{H_t + L_t + C_t}{3} \dots\dots\dots (2.13)$$

$$SM_t = \frac{\sum_{i=1}^n M_i}{n} \dots\dots\dots (2.14)$$

$$D_t = \frac{\sum_{i=1}^n |M_i - SM_t|}{n} \dots\dots\dots (2.15)$$

Keterangan :

H_t = high price pada waktu ke t
 C_t = closing price pada waktu ke t
 L_t = low price pada waktu ke t

Pada penelitian ini sepuluh indikator di atas akan digunakan untuk sebagai data input untuk model yang akan dikembangkan. Teknikal indikator tersebut digunakan karena berdasarkan jurnal (Armano et al., 2005; Huang & Tsai, 2009; Kim, 2003; Kim & Han, 2000; Yao, Chew, & Poh, 1999) menyatakan bahwa teknikal indikator merupakan nilai-nilai yang merepresentasikan masa depan yang diolah dari data masa lalu. Teknikal indikator biasa digunakan pakar saham untuk sebagai acuan dalam melakukan investasi pada suatu perusahaan. Namun analisa pada teknikal indikator dapat berbeda satu pakar dengan yang lainnya tergantung pengalaman dan pengetahuan.

Analisis saham dapat dilakukan dengan metode *chartist*. *Chartist* adalah metode proses analisis saham yang hanya melihat dari fitur dasar yaitu harga pembuka (*opening*) , harga terendah (*low*) , harga tertinggi (*high*) dan harga penutupan (*close*) (OHLC) (Faurina, 2019). *Chartist* biasa dipakai oleh pemula dalam bidang jual beli saham dengan melihat pergerakan *chart* harga yang menggambarkan *trend* pada bursa efek.

2.5 Data Mining

Teknologi *database* yang mengalami kemajuan secara masif sangat membantu berbagai aspek kehidupan yang berhubungan dengan data. Banyak bidang yang menghasilkan data setiap hari seperti bisnis, sosial, keilmuan, teknik, dan kesehatan. Data yang dulunya disimpan, diolah, dan dianalisis secara manual pada sekarang ini dikerjakan

dengan pendekatan teknologi. Lonjakan aliran data yang disebabkan oleh kemajuan teknologi ini tidak dapat dihindarkan. Namun aliran data yang masif ini akan menjadi tumpukan data yang tidak memberikan *knowledge* apabila tidak diolah dan dianalisis dengan pendekatan komputerisasi. *Data mining* merupakan proses untuk menemukan pola dan *knowledge* dari data yang sangat besar (Han, 2012). Dengan teknik *data mining* kumpulan data yang sangat besar dapat dianalisis dan diolah agar menjadi sebuah *knowledge* yang baru. Berikut adalah proses *data mining* secara umum (Han, 2012).

1. *Data Cleaning*

Proses *data cleaning* adalah membersihkan data dari elemen-elemen yang tidak relevan dan tidak perlu dalam proses penggalian *knowledge* data. Beberapa contoh dari data *cleaning* adalah menghapus baris data kosong dan menghapus beberapa fitur tidak penting atau tidak dibutuhkan oleh algoritma pengolah data seperti *machine learning*. Data pada masa ini masih banyak yang bersifat *incomplete*, *noisy*, and *inconsistent* sehingga proses data *cleaning* sebaiknya diterapkan agar data dapat digunakan dengan baik oleh algoritma pengolah data yang dipakai. Karena dalam melakukan analisa data yang baik salah parameter mutlak adalah data yang tidak mengandung nilai NaN, tidak *Redundant* dan lengkap dari segi kuantitas dan kualitas (Han, 2012).

2. *Data Integration*

Data Integration adalah proses penggabungan data dari beberapa sumber. *Data Integration* merupakan tahapan penggalian data yang sering diterapkan pada perusahaan yang ingin melakukan analisis data pada suatu bidang permasalahan dengan menggunakan beberapa sumber data yang berbeda. Proses *data integration* harus dilakukan dengan hati-hati agar tidak terjadi *redudancies* data dan *inconsistencies* data (Han, 2012).

3. *Data Selection*

Data selection adalah proses pemilihan data dari *database*. Proses *data selection* dilakukan apabila terdapat banyak pilihan dataset pada suatu database dengan kuantitas dan kualitas yang berbeda-beda.

4. *Data Transformation*

Data transformation adalah proses merubah bentuk data. Data yang diubah adalah nilai data dan transformasi fitur. Perubahan nilai data meliputi Normalisasi, Standarisasi, dan perubahan tipe data. Normalisasi dan Standarisasi merupakan proses *Scaling* nilai data agar tidak terjadi kecenderungan satu fitur atau lebih dalam mempengaruhi analisis. Perubahan tipe data biasanya terjadi pada data dengan tipe data Nominal. Data nominal merupakan data dengan nilai bukan angka yang memiliki identitas tersendiri. Contoh data nominal adalah data nama sekolah, data nama sekolah tidak dapat diukur dengan angka dan memiliki informasinya tersendiri. Data nominal biasanya diubah menjadi data biner yang diiringi dengan penambahan fitur yang biasa dikenal dengan *Dummy feature*. Transformasi fitur ada dua jenis yaitu Ekstraksi fitur dan reduksi fitur. Ekstraksi fitur adalah penambahan fitur dan reduksi fitur adalah pengurangan fitur.

5. *Data Mining*

Data Mining merupakan proses pengenalan pola pada data yang diberikan. Proses *data mining* menggunakan algoritma statistik atau *machine learning*. Algoritma yang digunakan akan menelusuri pola-pola yang terdapat pada suatu data sehingga akan didapatkan luaran *knowledge* dari algoritma tersebut. Khusus untuk *machine learning*, *knowledge* yang didapatkan berada dalam suatu model matematika.

6. *Pattern Evaluation*

Proses *Pattern Evaluation* adalah proses analisis sebuah *knowledge* yang telah didapatkan telah sama yang diinginkan atau belum. Dalam *machine learning* proses ini adalah analisis testing.

7. *Knowledge Presentation*

Proses *knowledge presentation* adalah proses visualisasi dan naratif dari suatu *knowledge*. Visualisasi dapat berupa tabel, bar, grafik, dan irisan pie. Naratif dapat berupa kalimat yang menceritakan *knowledge* dari pola yang didapatkan dari sekumpulan data. *Story telling* dari suatu *knowledge* data adalah suatu hal yang penting agar dapat dimengerti oleh semua orang dan memanfaatkan *knowledge* tersebut menjadi sebuah penimbang dalam pengambilan keputusan.

Praprosesing data pada bagian *data transformation* memiliki banyak metode yang berbeda-beda tergantung dari bentuk dan jenis data. Data pada penelitian ini adalah data *time series* yang memiliki tipe data numerik sehingga pada penelitian ini praprosesing menggunakan dua metode yaitu *min max scaler* sebagai metode normalisasi data dan *sliding windows*. Metode yang digunakan untuk melakukan *pattern evaluation* pada penelitian ini ada 3 yaitu *Mean Absolute Error* (MAE), *Root Mean Squared Error* (RMSE), dan *Directional Accuracy* (DA). Ketiga metode ini dipilih karena memiliki keunggulan dan kelemahannya masing-masing dalam mengevaluasi hasil prediksi. Berikut adalah penjelasan dari masing-masing metode *data transformation* dan *pattern evaluation* yang dipakai.

2.5.1 *Min Max Scaler*

Min max scaler adalah metode normalisasi yang menjaga hubungan antar fitur-fitur atau variabel-variabel memiliki kontribusi yang sama (Sugiartawan, Pulungan, & Sari, 2017). Variabel yang diukur pada skala yang berbeda tidak memberikan kontribusi yang sama pada model *fitting* & fungsi model learning dan mungkin akan menciptakan bias (Loukas, 2020). *Min max scaler* menjadi jawaban dari masalah ketidakseimbangan pengaruh variabel pada dataset dengan mengubah semua data pada dataset dengan skala yang sama yaitu antara 0 dan 1.

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)} \dots\dots\dots (2.16)$$

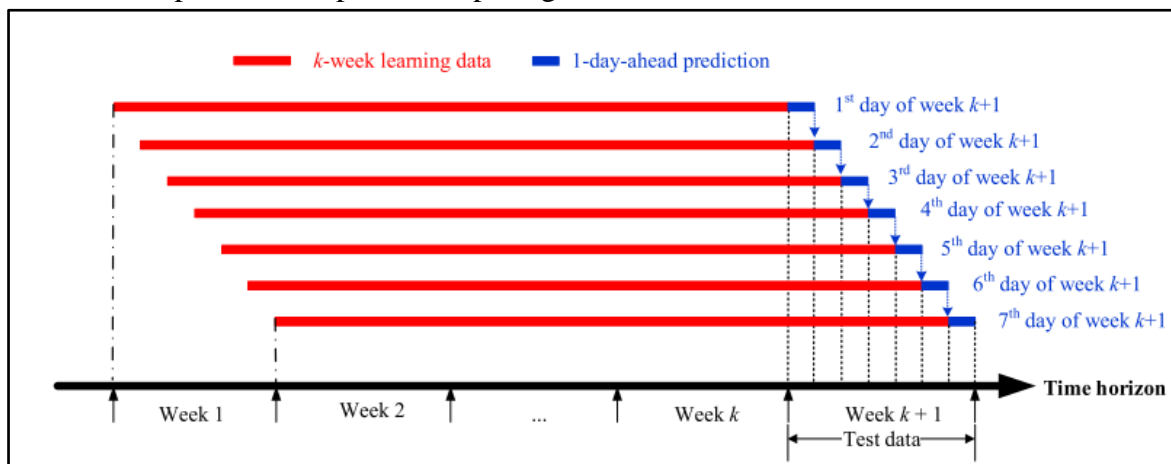
Keterangan :

x = data

2.5.2 *Sliding windows*

Sliding windows akan dipakai hanya pada data *chartist*. Data *technical indikator* tidak memakai metode ini karena sifat natural dari *technical indikator* sendiri adalah nilai-nilai yang

terkandung didalamnya adalah penggambaran dari data-data OHLC. *Sliding windows* adalah metode salah satu jenis metode pemrosesan konsep *drift* dalam aliran data agar data memiliki informasi dari suatu data pada rentang waktu tertentu yang telah ditentukan (Sugiartawan et al., 2017). Inti dari teknik ini adalah mekanisme pembaruan data. Aliran data (x) dibagi menjadi beberapa bagian blok data. Ketika jendela geser pindah ke blok berikutnya, blok baru ditambahkan ke jendela pada interval, dan blok terlama dihapus. Melalui metode pemilihan sampel dinamis ini, sampel untuk pemodelan diperbaharui. Ilustrasi dari proses ini dapat dilihat pada gambar 2.1.



Gambar 2.1 Ilustrasi *Sliding windows*

Sumber : Chou (2016)

2.5.3 Mean Absolute Error (MAE)

MAE adalah metode evaluasi yang sangat toleran terhadap outlier sehingga metode ini sangat cocok dipakai pada data yang sangat *noisy* (Chai & Oceanic, 2015). Metode ini tidak menghukum nilai error dengan sensitif sehingga bagus digunakan pada analisa data *non linear* dan *non stationary*. Rumusan MAE adalah sebagai berikut (Chai & Oceanic, 2015).

$$MAE = \frac{\sum_{i=1}^N |y_i - \hat{y}_i|}{N} \dots \dots \dots (2.17)$$

Keterangan :

\hat{y}_i = hasil prediksi ke i

y_i = hasil aktual ke i

2.5.4 Root Mean Squared Error (RMSE)

RMSE pada penelitian ini digunakan untuk mengukur performa model dalam mengenali data perusahaan tertentu. Karena setiap perusahaan memiliki *noisy* yang berbeda-beda. RMSE cocok dalam mengenali kualitas data suatu perusahaan karena sangat sensitif terhadap *outlier* karena RMSE akan menghukum dengan berat sehingga nilai error yang kecil akan semakin kecil dan nilai error yang besar akan semakin besar (Chai & Oceanic, 2015). *Outlier* pada data perusahaan merupakan hasil dari seberapa besar fluktuasi harga saham yang terjadi. Berikut adalah persamaan dari RMSE.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \dots \dots \dots (2.18)$$

Keterangan :

\hat{y}_i = hasil prediksi ke i

y_i = hasil aktual ke i

2.5.5 Directional Accuracy (DA)

DA mempunyai kemampuan mengukur performa model dalam mengukur arah *trend* atau fluktuasi. Standar industri keuangan dalam membangun suatu model prediksi adalah nilai DA dapat mencapai nilai akurasi 60% atau lebih. Berikut adalah persamaan dari DA. $a_t = 1$ apabila $(y_{t+1} - y_t) \times (\hat{y}_{t+1} - \hat{y}_t) \geq 0$ dan $a_t = 0$ jika kondisi lain terjadi.

$$DA = \frac{1}{N} \sum_{i=1}^N a_t \times 100\% \dots \dots \dots (2.19)$$

$$a_t = (y_{t+1} - y_t) \times (\hat{y}_{t+1} - \hat{y}_t) \dots \dots \dots (2.20)$$

Keterangan :

\hat{y}_t = hasil prediksi ke t

y_t = hasil aktual ke t

2.6 Deep Learning

Deep learning merupakan *machine learning* yang telah ditingkatkan dalam hal pengenalan pola dengan data dengan bentuk yang masih mentah. Dalam beberapa dekade *machine learning* melakukan klasifikasi dengan terlebih dahulu mengolah dan mentransformasikan data terlebih dahulu menjadi representasi internal yang sesuai dengan pola input yang dapat dijalankan oleh teknik *machine learning* (Lecun, Bengio, & Hinton, 2015). *Deep learning* merupakan pengembangan dari *machine learning* lebih lanjut yang mempunyai kemampuan dalam mengenali data tanpa praprosesing fitur yang kompleks. Praprosesing data yang kompleks tidak diperlukan dalam *deep learning* karena *deep learning* mempunyai kemampuan dalam mengenali fitur dan menentukan pengaruh dari fitur-fitur dalam data (Lecun et al., 2015).

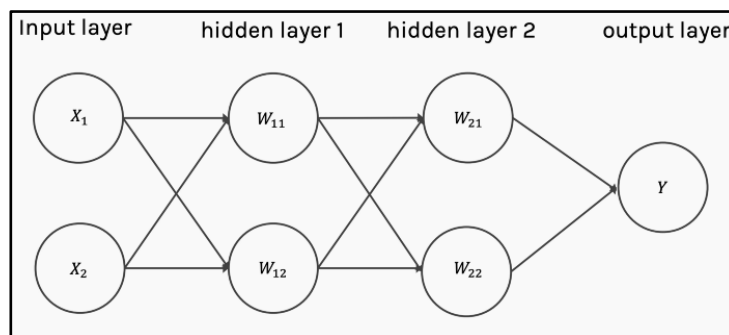
Deep Learning menjadi menjadi penyelesaian masalah untuk masalah yang kompleks seperti *visual object recognition* , *speech recognition* , *signal recognition* , dan *non linear time series* pada berbagai bidang seperti obat-obatan dan rekayasa gen. *Deep learning* menjadi trend dalam menyelesaikan masalah-masalah yang membutuhkan kemampuan dari sistem cerdas dalam mengenali pola kompleks suatu data (Lecun et al., 2015).

2.7 Artificial Neural Network (ANN)

Jaringan syaraf tiruan(JST) atau *Artificial Neural Network*(ANN) adalah model *machine learning* yang terinspirasi dari kerja otak manusia dalam memproses informasi dengan rangkaian struktur sel neuron. Sel neuron yang saling terhubung ini dapat mengenali pola kompleks dalam waktu yang singkat(Steward, 2019). Jaringan syaraf tiruan dapat memprediksi pola dengan informasi masa lalu. Jaringan syaraf mempunyai kemampuan

untuk memberi keputusan terhadap data yang belum pernah dipelajari sebelumnya (Agustin & Prahasto, 2012). ANN merupakan model yang mempunyai kemampuan untuk melakukan klasifikasi dan regresi pada dataset yang bersifat *nonlinear* dan *nonstationary*. Hal ini terbukti dengan kehandalan ANN dalam menangani berbagai data dengan *noise* yang tinggi. Model ANN memiliki kemampuan dalam mengenali pola pada data *time series* (Kara et al., 2011).

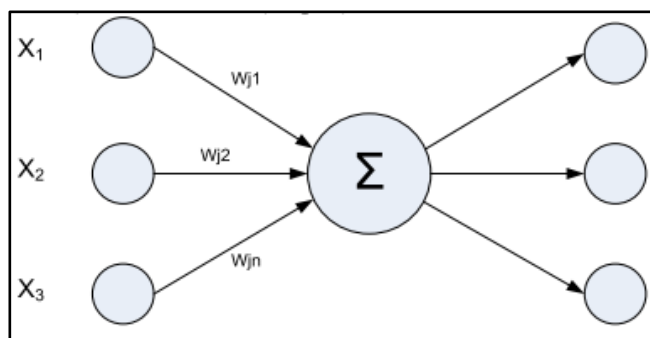
Arsitektur dari ANN adalah rangkaian tiruan *neuron* yang saling terhubung dengan input dan output. Di dalam *neuron* ini terdapat perhitungan kalkulus yang dapat mengenali pola suatu data. Model dari ANN terdiri dari tiga bagian yaitu *input* (masukan), *Hidden Layer* (lapisan tersembunyi), *output* (keluaran). *Input* dalam ANN adalah data masa lalu yang digunakan sebagai informasi yang akan dipelajari oleh model. *Output* dalam ANN adalah hasil prediksi dari kalkulasi yang dilakukan oleh model. *Hidden Layer* adalah rangkaian *neuron* yang mengkalkulasikan probabilitas dari setiap fitur input dalam mendapatkan nilai prediksi. Unit dari *hidden layer* tidak dapat diamati secara langsung (Agustin & Prahasto, 2012).



Gambar 2.2 Model ANN

Sumber: Steward (2019)

Rangkaian neuron dari gambar 2.2. memiliki input dua neuron, hidden layer 1 dua neuron, hidden layer 2 dua neuron, dan output satu neuron. Model ANN dapat menghitung probabilitas dari input dengan cara setiap input mengirimkan sinyal berupa nilai input dan bobot (weight) yang dilambangkan dengan huruf "W" ke setiap neuron yang ada. Kemudian setiap neuron akan mengaktifkan fungsi penjumlahan. Setelah itu neuron akan mengaktifkan fungsi aktivasi. Setelah itu akan didapat sebuah nilai yang kemudian akan dijadikan probabilitas untuk layer selanjutnya. Nilai probabilitas ini akan jadi bobot untuk layer selanjutnya (Agustin & Prahasto, 2012).



Gambar 2.3 Model neuron

Sumber : (Agustin & Prahasto, 2012)

Berikut adalah persamaan dari keluaran setiap *neuron* :

$$Y_j = \sigma(\sum_{i=0}^n X_i W_i + \beta_j) \dots\dots\dots (2.21)$$

Keterangan :

Y_j = keluaran dari *neuron*

σ = fungsi aktivasi (pada persamaan ini fungsi *sigmoid*)

X = nilai dari input

W = bobot

B = bias

Setiap bobot dan bias pada *neuron* berbeda. Nilai-nilai dari bobot, input, dan bias inilah yang akan menjadi perhitungan probabilitas setiap input maupun hubungan intuitif setiap input. Bias pada formula 2.11 merupakan faktor koreksi yang berfungsi terhadap kelengkapan variabel-variabel yang telah kita tetapkan. Apabila pada saat pelatihan nilai bobot dari bias menjadi sangat besar dibandingkan dengan nilai bobot variabel input, maka dapat disimpulkan bahwa masih terdapat variabel input lainnya yang cukup menentukan output (Steward, 2019).

Fungsi aktivasi merupakan suatu fungsi berupa suatu persamaan matematis yang memungkinkan ANN dapat mengenali pola data atau memberikan hasil perhitungan dari data input yang belum pernah dipelajarinya. Fungsi aktivasi memiliki beberapa macam persamaan tergantung masalah yang akan diselesaikan (Karlik & Olgac, 2011). Beberapa jenis fungsi aktivasi adalah *Rectified linear unit*(ReLU), *Sigmoid*, identitas, *Binary Step*, *Logistic*, TanH, *Softmax*, dll. Berikut beberapa rumus dari fungsi aktivasi :

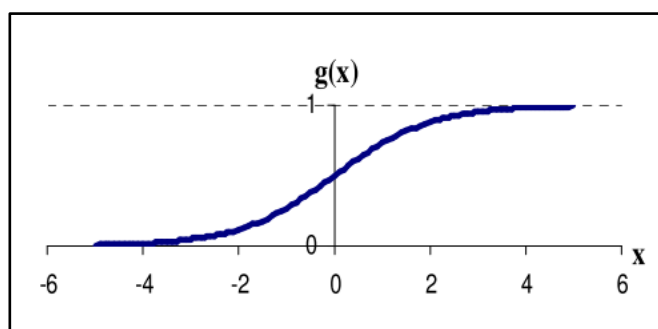
1. Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}} \dots\dots\dots (2.22)$$

Keterangan :

x = data *input*

e = konstanta matematika (2,718281828459045235360287471352)



Gambar 2.4 Fungsi Sigmoid
Sumber : Karlik & Olgac (2011)

2. TanH

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \dots\dots\dots (2.23)$$

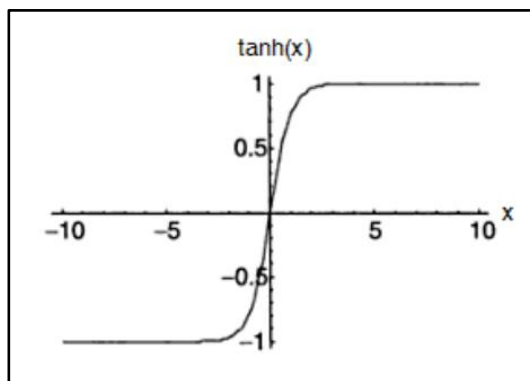
Keterangan :

x = data *input*

ϵ = konstanta matematika (2,718281828459045235360287471352)

Sinh = sinus hiperbolik

Cosh = cosinus hiperbolik



Gambar 2.5 Fungsi TanH

Sumber : Karlik & Olgac (2011)

3. Rectified linear unit(ReLU)

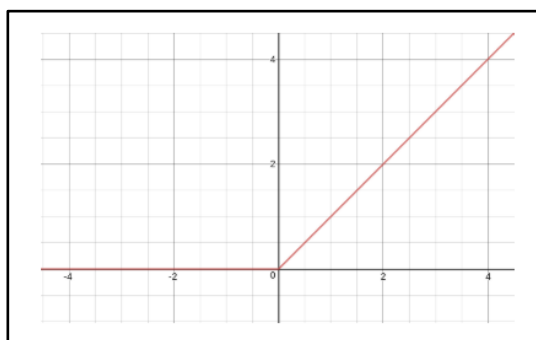
$$f(x) = \max(0, x) \dots\dots\dots (2.24)$$

$$f(x) = \begin{cases} 0, & \text{jika } x \leq 0 \\ x, & \text{jika } x > 0 \end{cases} \dots\dots\dots (2.25)$$

Keterangan :

x = data *input*

max = maksimal



Gambar 2.6 Fungsi ReLU

Sumber : Agarap (2018)

4. Leaky ReLU

$$f(x) = \max(ax, x) \dots\dots\dots (2.26)$$

$$f(x) = \begin{cases} x, & \text{jika } x \geq 0 \\ ax, & \text{jika } x < 0 \end{cases} \dots\dots\dots (2.27)$$

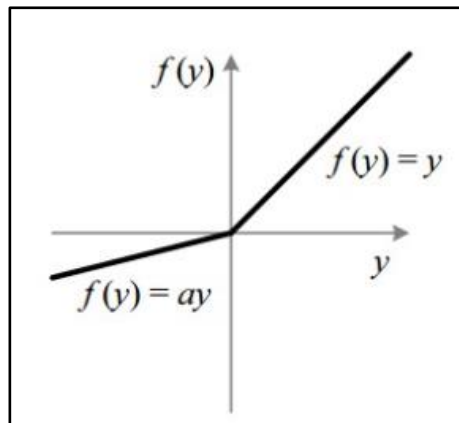
Keterangan :

x = data *input*

max = maksimal

a = konstanta Leaky ReLU (0.01)

Konstanta Leaky ReLU adalah 0.01, apabila konstanta nya selain 0.01 fungsi itu biasa disebut *Randomized ReLU*.

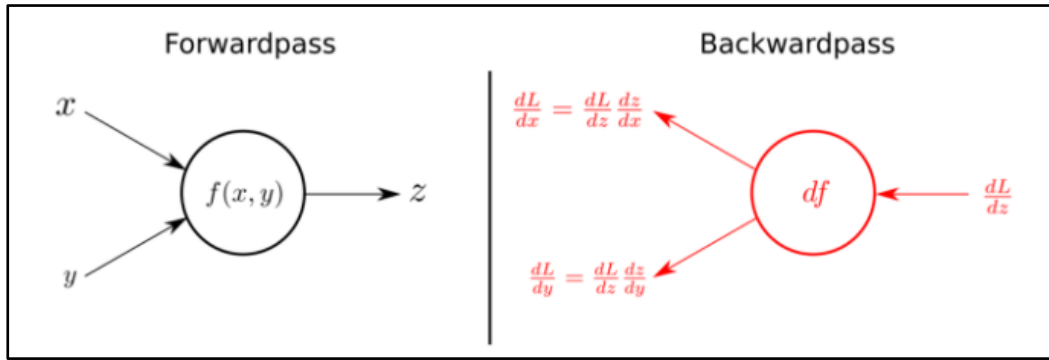


Gambar 2.7 Fungsi Leaky ReLU
Sumber : Setiaji (2018)

2.8 Gradient Descent

Untuk mendapatkan hasil yang maksimal dari Model ANN maka bobot dan bias memerlukan proporsi yang seimbang dan rasional terhadap hubungan antara probabilitas setiap input. Namun perhitungan probabilitas pada setiap parameter yang ada pada model ANN akan sangat memakan waktu dan sumber daya apabila model yang dikembangkan sangat kompleks. Sebagai contoh pada kasus deteksi gambar yang memiliki ribuan input yang berasal dari setiap piksel gambar. Pada kasus deteksi gambar juga diperlukan lebih dari satu *Hidden Layer* yang memiliki banyak *neuron*. Untuk mengatasi masalah ini maka diperlukan sebuah algoritma yang bernama *Gradient Descent*.

Gradient Descent adalah algoritma yang berfungsi untuk melakukan *update* pada semua parameter ANN dengan melakukan kalkulasi *derivatives* (turunan) dengan meninjau *gradient* terhadap *loss function* di ANN. *Loss function* adalah fungsi error pada saat pelatihan yang mana akan menjadi tolak ukur dalam mencari *gradient* setiap parameter. Pada kasus prediksi yang membutuhkan ketidaksensitifan terhadap outlier seperti saham dan finansial keuangan *loss function* yang digunakan adalah MAE. Berikut persamaan *loss function* MAE. Peninjauan *gradient* ini dilakukan secara *backpropagation*. *Backpropagation* (BP) adalah proses ANN dalam kalkulasi dari *loss function* ke *input* atau dengan sederhana disebut sebagai jaringan terbalik. Proses *backpropagation* bisa dilihat pada gambar 2.8. Turunan pada *gradient* ini dimaksudkan untuk mencari seberapa sensitif bobot dan bias terhadap *output* yang dihasilkan. *Gradient descent* adalah algoritma yang mengukur kecondongan atau kemiringan bobot dan bias. *Loss function* merupakan fungsi yang menilai error model ANN (Steward, 2019). Pada model ANN inisiasi pertama parameter bobot dan bias adalah acak. Setelah itu *Gradient Descent* akan melakukan *update* pada setiap parameter. Proses ini akan terus dilakukan berulang ulang pada setiap dataset sampai konvergen dan mencapai *minimum loss function*. *Minimum loss function* artinya model memiliki error yang paling rendah.



Gambar 2.8 Backpropagation

Sumber : Agarwal (2017)

$$C = |a^{(L)} - y| \dots\dots\dots (2.28)$$

$$a^{(L)} = \sigma(z^{(L)}) \dots\dots\dots (2.29)$$

$$z^{(L)} = w^{(L)}a^{(L-1)} + b^{(L)} \dots\dots\dots (2.30)$$

$$\frac{\partial C}{\partial a^{(L)}} = \frac{a^{(L)} - y}{|a^{(L)} - y|} \dots\dots\dots (2.31)$$

$$\frac{\partial a^{(L)}}{\partial z^{(L)}} = \sigma'(z^{(L)}) \dots\dots\dots (2.32)$$

$$\frac{\partial z^{(L)}}{\partial w_i} = a^{(L-1)} \dots\dots\dots (2.33)$$

$$\frac{\partial C}{\partial w_i} = \frac{\partial C}{\partial a^{(L)}} \frac{\partial a^{(L)}}{\partial z^{(L)}} \frac{\partial z^{(L)}}{\partial w_i} \dots\dots\dots (2.34)$$

$$\nabla C(w_i) \equiv \left[\frac{\partial C}{\partial w_i}, \frac{\partial C}{\partial w_{i+1}}, \frac{\partial C}{\partial w_{i+2}}, \frac{\partial C}{\partial w_{i+3}}, \dots, \frac{\partial C}{\partial w_{i+n}} \right] \dots\dots\dots (2.35)$$

$$\Delta w_i = -\eta \nabla C(w_i) \dots\dots\dots (2.36)$$

$$w_{i+1} = w_i + \Delta w_i \dots\dots\dots (2.37)$$

Keterangan :

C = cost function (persamaan 2.33 memakai MAE cost function)

$a^{(L)}$ = nilai fungsi aktivasi layer ke L atau output (L bukan eksponen)

$a^{(L-1)}$ = nilai aktivasi layer ke L-1

L = identitas layer

σ = fungsi aktivasi (pada kasus ini fungsi sigmoid)

$z^{(L)}$ = fungsi feed forward

w_i = bobot ke i

$\nabla C(w_i)$ = gradient terhadap bobot

Δw_i = seberapa banyak perubahan bobot

η = learning rate

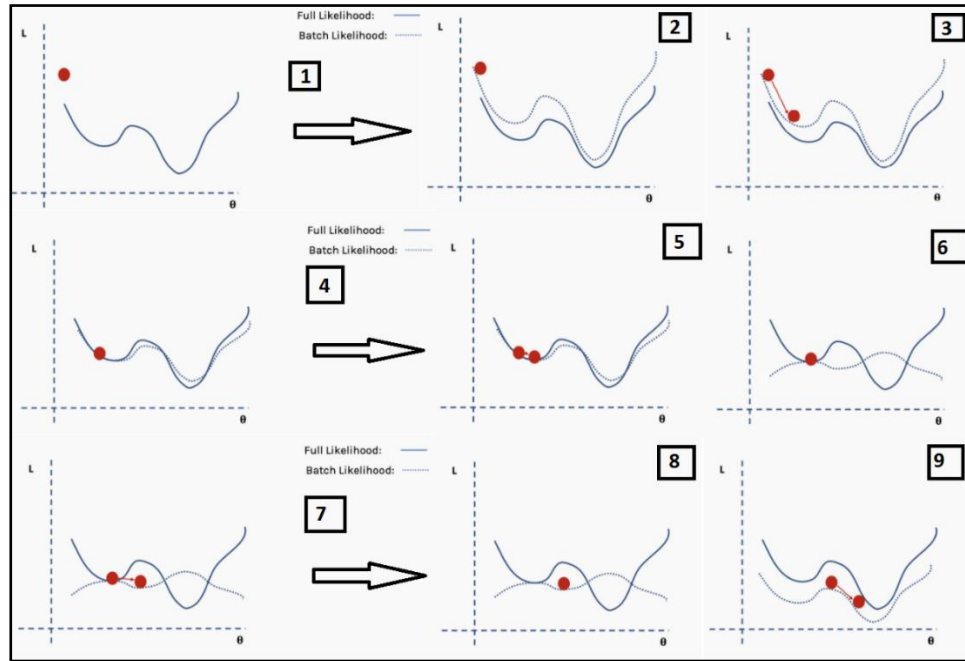
w_{i+1} = bobot baru

Persamaan 2.29 sampai 2.38 adalah serangkaian contoh persamaan yang digunakan ketika menggunakan algoritma *gradient descent* ketika optimasi bobot pada ANN (Nielsen, 2019). *Gradient descent* menggunakan metode turunan *chain rule*. Persamaan diatas adalah

contoh untuk optimasi bobot. Persamaan diatas berlaku untuk melakukan optimasi pada bias dengan hanya mengganti w_i dengan b . Persamaan diatas dapat diterapkan pada semua metode ANN dengan patokan turunan *cost function* diturunkan terhadap bobot atau bias yang ingin dicari. Persamaan 2.29 merupakan *cost function* MAE. Persamaan 2.29 dapat dirubah seiring dengan permasalahan yang dihadapi. Apabila permasalahan yang dihadapi adalah *binary* atau klasifikasi *multiclass* maka fungsi 2.29 yang dipakai adalah *Binary Cross-Entropy* atau *Multi-Class Cross-Entropy Loss*. Pada persamaan 2.36 semua nilai dari turunan *cost function* terhadap bobot dikumpulkan agar dapat dipakai pada persamaan selanjutnya. Persamaan 2.37 mengalikan negatif *learning rate* terhadap seluruh *gradient* yang ada. Kemudian persamaan 2.38 melakukan *update* terhadap seluruh bobot.

Gradient descent memiliki kelemahan pada saat terjebak di *local optima*. Solusi yang dapat dipakai untuk menghindari *local optima* adalah dengan memakai metode *batch* dengan *stochastic gradient descent*. *Batch* merupakan proses pelatihan dengan membagi dataset menjadi beberapa bagian kemudian dilatih terpisah dengan *error surface* yang berbeda. Ilustrasi batch terdapat pada gambar 2.9. *Error surface* pada masing-masing *batch* data berbeda dengan dengan *error surface* data keseluruhan. Hal ini membuat solusi akan mencari nilai minimum pada setiap *batch* (Steward, 2019). Keterangan tahapan pada gambar 2.9 adalah sebagai berikut :

1. Gambar pertama menunjukkan grafik *error surface* secara keseluruhan dan lingkaran merah adalah solusinya.
2. Kemudian dataset kita bagi menjadi beberapa bagian atau subset.
3. Gambar kedua adalah bagaimana *error surface* yang dihasilkan oleh subset pertama.
4. Gambar ketiga menggambarkan bagaimana solusi turun pada *loss function* terendah.
5. Gambar keempat menggambarkan *error surface* dari subset kedua. Gambar kelima mengilustrasikan solusi turun pada *loss function* terendah pada *error surface* subset tersebut.
6. Gambar keenam adalah *error surface* subset ketiga.
7. Gambar ketujuh mengilustrasikan solusi turun pada *loss function* terendah pada *error surface* subset tersebut.
8. Gambar kedelapan menggambarkan *error surface* dari subset kedua.
9. Gambar kesembilan mengilustrasikan solusi turun pada *loss function* terendah pada *error surface* subset tersebut, pada tahap ini *loss function* menunjukkan turun pada *global optima*.



Gambar 2.9 Ilustrasi batch

Sumber : Steward (2019)

Salah satu kekurangan *gradient descent* adalah memiliki nilai learning rate yang tetap sehingga learning rate menjadi parameter yang sangat fatal apabila salah inisialisasi. Perkembangan selanjutnya dari *gradient descent* adalah algoritma dengan learning rate yang adaptif atau dapat berubah-ubah menyesuaikan dengan *loss function* dengan harapan algoritma pelatihan dapat menyesuaikan keadaan model dan menghasilkan nilai eror yang lebih kecil. Salah satu algoritma adaptif *learning rate* dengan tingkat performa bagus adalah *Adaptive Moment Estimation* (ADAM). ADAM merupakan metode yang sedikit lebih unggul dalam performa dibandingkan algoritma adaptif *learning rate* lainnya seperti *Root Mean Square Propagation* (RMSProp) dan *Adaptive Gradient Algorithm* (AdaGrad) (Chandra, 2019). Berikut adalah persamaan dari algoritma ADAM (Bock, Goppold, & Weiß, 2018)

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla C(w_t) \dots \dots \dots (2.38)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) (\nabla C(w_t))^2 \dots \dots \dots (2.39)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \dots \dots \dots (2.40)$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \dots \dots \dots (2.41)$$

$$w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \dots \dots \dots (2.42)$$

Keterangan :

β_1 = parameter *decay rate* 1

β_2 = parameter *decay rate* 2

m_t = estimasi momen 1

v_t = estimasi momen 2

$\nabla C(w_i)$ = *gradient* (pada contoh rumus gradient terhadap bobot)

\hat{m}_t = *bias correction* pada estimasi momen 1

v_t = *bias correction* pada estimasi momen 2

w_{t+1} = bobot baru

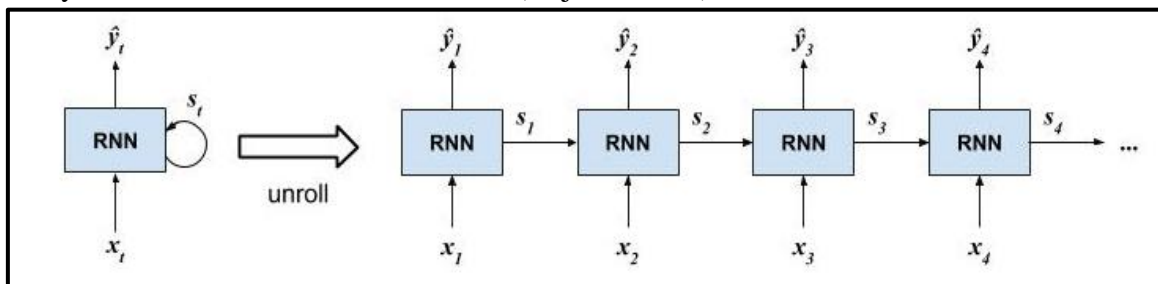
w_t = bobot lama

η = *learning rate*

2.9 Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNN) adalah arsitektur dari ANN yang pelatihannya dipanggil berulang ulang. RNN termaksud ke dalam *deep learning* karena arsitekturnya yang memakai banyak *layer*. RNN mengalami perkembangan yang sangat pesat setelah menyelesaikan berbagai masalah di dunia industri. RNN adalah model yang dapat menyelesaikan masalah pemrosesan bahasa alami atau *natural language processing* (NLP), pengenalan suara, pengenalan suara, sintesa, musik, data *time series*, analisa video, analisa deret DNA, dll (Priyono, 2018). Model RNN mampu menyelesaikan berbagai masalah data sekuensial, yaitu data yang mempunyai urutan konstan dan terkait. Beda dengan ANN *feedforward* biasa yang menganggap data yang dilatih adalah data *independent, identically dan distributed* (IID). ANN biasa juga tidak memiliki memori untuk sehingga data akan dilatih tanpa ada konsiderasi terhadap sampel-sampel sebelumnya (Priyono, 2018).

RNN adalah arsitektur ANN yang memproses input secara sekuensial. *Output* dari satu *state* waktu RNN akan menjadi salah satu input bagi *state* RNN selanjutnya. Dengan adanya masukan input dari *state* sebelumnya maka RNN dapat mempertimbangkan *ouput* dengan melihat *output* sebelumnya. Pengulangan input inilah yang membuat RNN dapat menyelesaikan masalah data sekuensial (Priyono, 2018).



Gambar 2.10 Arsitektur RNN

Sumber : Priyono (2018)

Pada RNN terdapat parameter U, V dan W. Untuk setiap langkah waktu t terdapat kalkulasi state S_t dari *input* X_t dan *state* sebelumnya, masing-masing dikalikan dengan parameter U dan W lalu diproses dengan fungsi aktivasi *tanh*. Dari S_t kemudian dikalkulasi output Y_t dengan cara mengalikan dengan parameter V dan melewati pada fungsi aktivasi *softmax* (Priyono, 2018).

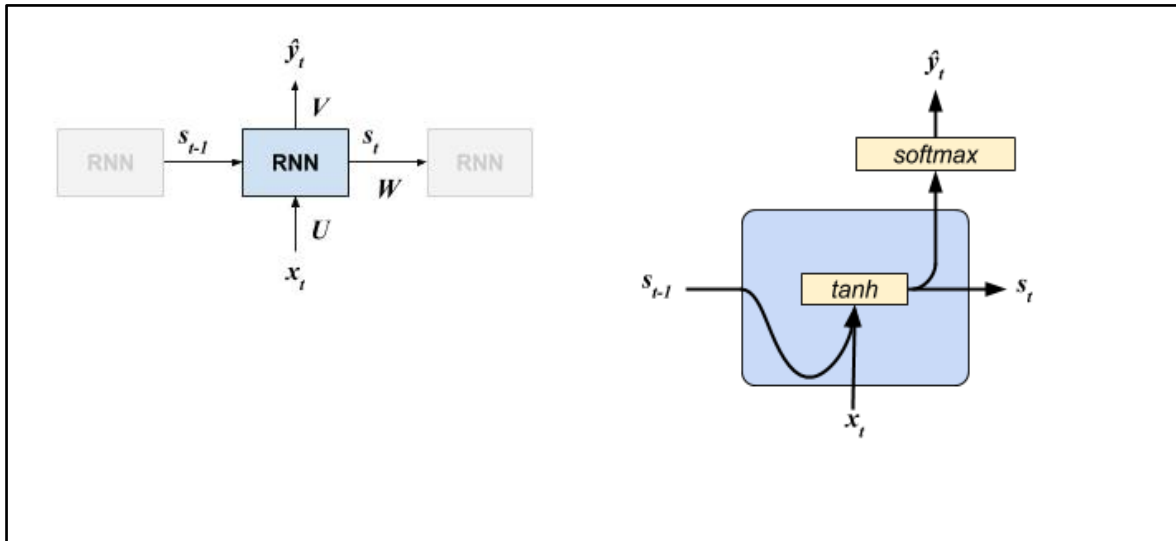
$$S_t = \tanh(U * X_t + W * S_{t-1}) \dots \dots \dots (2.43)$$

$$Y_t = \text{softmax}(V * S_t) \dots \dots \dots (2.44)$$

Keterangan :

S_t = *state* ke t

S_{t-1} = state sebelum t
 X_t = input ke t
 Y_t = output ke t
 \tanh = fungsi aktivasi *tanh*
 U = parameter U
 V = parameter V
 W = parameter W



Gambar 2.11 Detail kerja sel RNN

Sumber : Priyono (2018)

Pelatihan RNN bertujuan untuk menemukan nilai V , W , dan U agar memiliki *loss function* yang rendah atau minimum. Untuk mendapatkan nilai optimal dari parameter tersebut maka algoritma yang biasa dipakai adalah *gradient descent* (SGD) yang mana dilakukan secara *backpropagation* (BP). Secara umum, RNN juga melakukan BP, namun ada hal yang khusus. Parameter U , V , dan W (terutama U dan W) melakukan kalkulasi dari *state* waktu sebelumnya, maka kalkulasi gradien pada langkah waktu t , mesin harus menghitung turunannya pada langkah waktu $t-1$, $t-2$, $t-3$, dan seterusnya sampai titik awal ($t=1$). Metode ini dinamakan *backpropagation through time* (BPTT) (Priyono, 2018).

RNN merupakan model *machine learning* yang termaksud kedalam *deep learning* karena model ANN ini memiliki banyak lapisan *layer*. Lapisan ANN yang banyak dapat membuat masalah pada waktu pelatihan yang dinamakan *Vanishing Gradient*. *Vanishing Gradient* adalah sebuah situasi dimana *gradient* yang timbul karena perubahan *output* semakin lama semakin mengecil hingga mendekati nol ketika melakukan *backpropagation*, *gradient* tersebut dapat dikatakan “menghilang” ketika sampai pada lapisan awal ANN (Priyono, 2018). Peristiwa ini lebih buruk terjadi pada RNN yang melakukan BPTT. RNN akan mengalami *vanishing gradient* ketika *frame* waktu pada data input sangat panjang. Ini disebabkan karena RNN melakukan BPTT yang sangat panjang hingga data pada waktu pertama. *Vanishing gradient* disebabkan oleh sifat perkalian antar bilangan pecahan.

Perkalian pecahan secara eksponensial akan membuat nilai yang dikalikan semakin lama semakin mengecil namun tidak nol, namun mendekati nol (Priyono, 2018).

ANN dengan lapisan yang banyak juga dapat mengalami *exploding gradient*. *Exploding gradient* adalah situasi dimana *gradient* yang timbul karena perubahan *output* sangat besar. Masalah ini mudah ditangani dengan menerapkan suatu batas atas untuk nilai *gradient* (Priyono, 2018). Untuk mengatasi masalah *Vanishing Gradient* solusi yang paling sering digunakan adalah menggunakan unit RNN yang lain seperti *Gated Recurrent Unit* (GRU). Ada beberapa cara lain untuk memecahkan masalah *Vanishing Gradient*, yaitu inisialisasi parameter yang benar, regularisasi, dan penggunaan ReLU untuk mengganti fungsi aktivasi tanh atau sigmoid.

2.10 Gated Recurrent Unit (GRU)

Gated Recurrent Unit (GRU) adalah model RNN yang telah dikembangkan dalam mengatasi masalah *long memory dependency* ketika menghadapi data dengan *frame* waktu yang panjang karena RNN akan mengalami *Vanishing Gradient* ketika melakukan BTT (Zhang & Kabuka, 2018). GRU merupakan pengembangan dari metode *Long Short Term Memory* (LSTM). Kelebihan dari GRU adalah bobot dan parameter yang di latih lebih sedikit dari LSTM sehingga kecepatan pelatihan model lebih cepat dari pada LSTM dan implementasi yang lebih sederhana dari LSTM (Struye & Latré, 2020). Bobot latih dan parameter GRU lebih sedikit daripada LSTM karena *Gate* atau gerbang GRU lebih sedikit daripada LSTM. LSTM memiliki tiga gerbang yaitu *forget gate* (f_t), *input gate* (i_t), dan *output gate* (O_t). GRU menggabungkan gerbang *forget* dan *input* menjadi *update gate* (z). Gerbang kedua dari GRU adalah *reset gate* (r). Apabila *ouput* yang dihasilkan LSTM ada dua nilai yaitu *hidden state* dan *cell state* maka *Output* dari GRU hanya satu yaitu *hidden state* (Chen, Jiang, & Zhang, 2019).

GRU mempunyai dua gerbang yaitu gerbang *update* dan gerbang *reset*. Gerbang *update* bertanggung jawab untuk memperbarui memori jaringan yang sedang dilatih. Gerbang *update* memungkinkan jaringan untuk mengingat input data tertentu berdasarkan *hidden state* sebelumnya. Semakin besar nilai *update* maka informasi yang dibawa oleh *state* sebelumnya akan semakin besar. Gerbang *reset* bertanggung jawab untuk menghapus memori jaringan saat pelatihan yang memungkinkan jaringan untuk melupakan nilai-nilai tertentu. Semakin besar nilai *reset* maka informasi yang dibawa dari *state* sebelumnya semakin banyak yang diabaikan. Persamaan transisi dalam *hidden state* GRU adalah sebagai berikut (Chen et al., 2019).

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \dots \dots \dots (2.45)$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \dots \dots \dots (2.46)$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h (r_t \odot h_{t-1}) + b_h) \dots \dots \dots (2.47)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \dots \dots \dots (2.48)$$

Keterangan :

z_t = *update gate*

r_t = *reset gate*

\tilde{h}_t = *candidates value*

persamaan 2.48 disebut Hidden states. GRU tidak memiliki Cell State seperti LSTM. Hidden State di dalam arsitektur GRU merupakan penyederhanaan dari dua memori sel LSTM yaitu Cell State dan Hidden State. Hidden State merupakan pembawa informasi bagi state selanjutnya agar terjadi pembaharuan informasi terhadap paramete-parameter di dalam modul GRU (Chen et al., 2019).

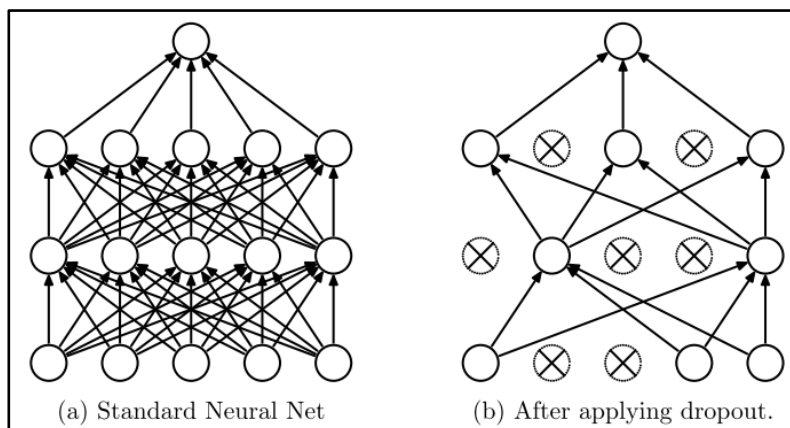
2.11 Parameter dan Arsitektur

GRU melakukan optimasi menggunakan algoritma *Gradient descent*. Pada ANN yang menggunakan *gradient descent* jumlah *epoch* dan jumlah *batch* adalah parameter yang perlu diinisialisasi. *Epoch* adalah suatu nilai yang menentukan seberapa banyak jaringan ANN akan dilatih dengan data yang sama. *Batch* adalah nilai jumlah *mini batch* pada saat training. Fungsi batch telah dijelaskan pada sub bab 2.8 . Jumlah *hidden layer* merupakan *hyperparameter* ANN. Penentuan dari *epoch*, *mini batch*, dan *hidden layer* tidak mempunyai aturan baku dan cenderung ditentukan dengan metode *trial and error* (Khoolish, 2017).

Model RNN-GRU satu *hidden layer* disebut dengan Shallow RNN-GRU. Stacked LSTM merupakan pengembangan Shallow RNN-GRU yang memiliki beberapa hidden layer. Penumpukan *hidden layer* pada RNN-GRU membuat model RNN-GRU menjadi lebih dalam (*deep*). RNN-GRU dapat mempelajari pola-pola detail yang terdapat pada data. Namun tidak selamanya *stacked* adalah arsitektur yang dapat menyelesaikan semua masalah dengan sempurna. Beberapa kasus *Shallow* dapat lebih baik daripada *Stacked*.(Faurina, 2019)

2.12 Dropout

ANN dalam dengan banyak parameter merupakan sistem pembelajaran mesin yang *powerfull*. Salah satu isu ANN adalah *overfitting*, *overfitting* merupakan keadaan dimana jaringan ANN sangat (*over*) mengenali (*fit*) pola data *training* sehingga jaringan ANN tidak dapat memprediksi data baru dengan pola yang baru (Srivastava et al. 2014). *Dropout* adalah teknik untuk mengatasi masalah *overfitting*. Ide utama dari *dropout* adalah melepaskan unit secara acak (bersama dengan koneksinya) dari jaringan ANN selama pelatihan. Metode dapat diilustrasikan pada gambar 2.16. Hal ini mencegah unit terlalu banyak beradaptasi bersama. Selama pelatihan, sampel *dropout* berasal dari sejumlah eksponensial jaringan "tipis" yang berbeda. Pada waktu pengujian, mudah untuk memperkirakan efek rata-rata prediksi semua jaringan yang ditipiskan ini hanya dengan menggunakan satu jaringan tak berujung yang memiliki bobot lebih kecil. *Dropout* secara signifikan mengurangi *overfitting* dan memberikan peningkatan besar dibandingkan metode regularisasi lainnya (Srivastava et al. 2014).



Gambar 2.16 Ilustrasi dropout

Sumber : Srivastava et al (2014)

2.13 Penelitian Terdahulu

Dasar dari penelitian ini yaitu adalah prediksi data *time series* yang memiliki sifat *nonlinear*, *nonstationary*, dan *long memory dependency* menggunakan metode RNN dengan unit GRU. Penelitian ini meninjau literatur yang berkaitan dengan data *time series* dengan sifat *nonlinear*, *nonstationary*, dan *long memory dependency* sebagai acuan. Penelitian ini juga meninjau literatur arsitektur RNN dengan unit lain seperti LSTM dan RNN konvensional sebagai patokan dalam menganalisis performa arsitektur RNN. Penelitian ini juga meninjau literatur lain yang berkaitan dengan data *time series* dengan metode selain RNN sebagai pembandingan metode.

Tabel 2.1 Tabel Penelitian Sebelumnya

No	Judul	Penulis	Metode	Hasil
1.	Klasifikasi Pergerakan Harga Saham Jangka Pendek Menggunakan Principal Component Analysis dan Jaringan Long Short Term Memory : Studi Kasus Pada Saham Bursa Efek Indonesia	Ruvita Faurina (2019)	Principal Component Analysis dan Jaringan Long Short Term Memory	Hasilnya adalah berbagai model yang dibuat dengan banyak pendekatan <i>praprocessing</i> dan optimasi pada arsitektur LSTM tanpa dan dengan PCA. PCA terbukti dapat meningkatkan akurasi dari LSTM sebesar 8,21%.
2.	Short-term Traffic Flow Prediction Using a Methodology Based on ARIMA and RBF-ANN	Kui, Lin li Chun, Jie Zai Jian, Min Zu (2017)	ARIMA dan RBF-ANN	Hasilnya adalah ARIMA mempunyai MAE sebesar 17,34 , MRE(%) sebesar 11,28 dan MSE sebesar 1,41. Hasil RBF-ANN adalah MAE sebesar 16,30 , MRE(%) sebesar 10,15 dan MSE sebesar 1,36. Dari hasil tersebut terlihat RBF-ANN unggul dalam akurasi
3.	Predicting direction of stock price index movement using artificial neural networks and support vector machines:	Kara, Yakup Acar Boyacioglu, Melek Baykan, Ömer Kaan (2011)	Artificial Neural Networks dan Support Vector Machines	Hasil penelitian ini berupa perbandingan akurasi prediksi harga saham kedua model. Hasilnya akurasi BPNN lebih baik dari pada SVM. Rata-rata performa akurasi yang

Tabel 2.2 Tabel Penelitian Sebelumnya (lanjutan)

No	Judul	Penulis	Metode	Hasil
	The sample of the Istanbul Stock Exchange			menggunakan metode T-test adalah <i>Neural network</i> (75,74%) lebih baik ketimbang SVM (71,52%) Penulis juga menekankan hasil penelitian tergantung dua hal yaitu rancangan model dan input variabel
4.	Using artificial neural network models in stock market index prediction	Guresen, Erkam Kayakutlu, Gulgun Daim, Tugrul U. (2011)	<i>multi-layer perceptron</i> (MLP), <i>dynamic artificial neural network</i> (DAN2) dan hybrid neural networks yang menggunakan <i>generalized autoregressive conditional heteroscedasticity</i> (GARCH)	Hasil dari penelitian ini adalah perbandingan akurasi dari hasil prediksi harga saham dari ketiga algoritma. Dalam penelitian ini MLP memiliki hasil yang lebih bagus daripada dua algoritma lainnya walaupun hanya terdapat sedikit perbedaan. Hasil uji MLP mempunyai MSE sebesar 2478,1468 , MAD sebesar 41,153 , MAD% sebesar 2,516. Hasil GARCH-MLP mempunyai MSE sebesar 3665,8387, MAD sebesar 42,739 , MAD% sebesar 2,775. Hasil DAN2 mempunyai MSE sebesar 1472,278, MAD sebesar 32,875, MAD% sebesar 2,768.
5	Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast	Ma, Tao Antoniou, Constantinos Toledo, Tomer (2020)	<i>Neural Network</i> , ARIMA, Dan SVR	Penelitian ini menggabungkan metode statistik dengan metode <i>machine learning</i> . Model yang dibandingkan adalah <i>Neural network</i> – ARIMA (NN-ARIMA) dengan NN <i>conventional</i> untuk mengukur seberapa jauh pengaruh ARIMA dalam analisis. Hasilnya adalah perbandingan error yang diukur dengan MSE kemudian error kedua metode (NN-ARIMA dan NN) kemudian dicari <i>error reduction</i> NN-ARIMA terhadap NN. Rata-rata <i>error reduction</i> rentang prediksi satu bulan adalah (13.40) , tiga bulan (8,9) dan enam bulan (10,43).
6.	A Multi Hidden Recurrent Neural Network with a modified Grey Wolf Optimizer	Id, Tarik A Rashid Abbas, Dosti K Turel, Yalin K	RNN dan <i>Modified Grey Wolf Optimizer</i>	Hasilnya adalah perbandingan dari dua metode yaitu RNN dan MLP kemudian dioptimasi dengan <i>Grey Wolf Optimizer</i> dalam melakukan klasifikasi kelemahan siswa yang kemudian di tes dengan <i>cross validation</i> M-RNNGWO (98%) dan RNNGWO(94%).

Tabel 2.3 Tabel Penelitian Sebelumnya (lanjutan)

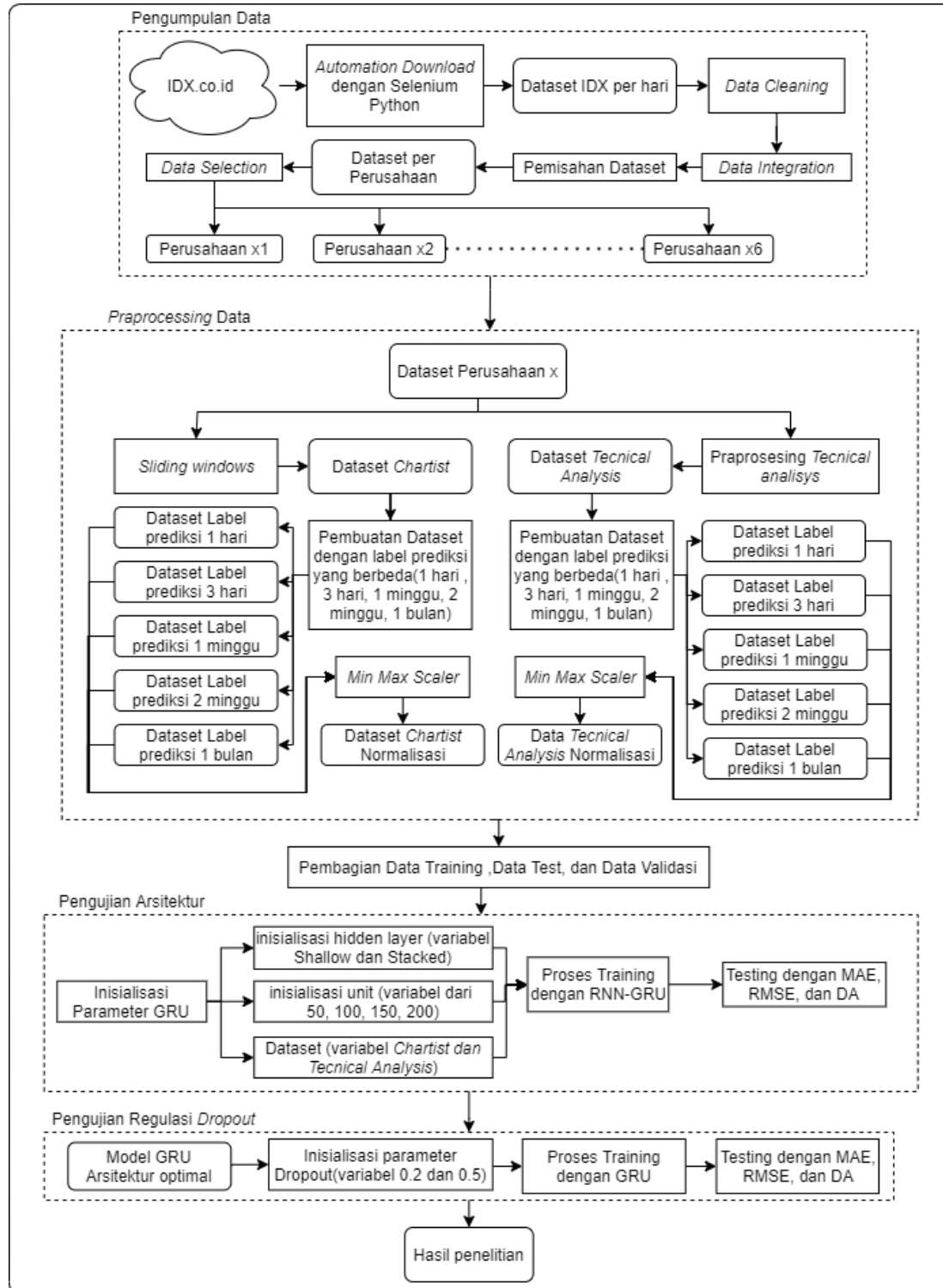
No	Judul	Penulis	Metode	Hasil
7.	Energy Load Forecasting using Deep Learning Approach- LSTM and GRU in Spark Cluster	Kumar, Sumit Hussain, Lasani Banarjee, Sekhar Reza, Motahar (2018)	LSTM, GRU, dan Standard RNN	Hasilnya adalah perbandingan akurasi prediksi pemakaian energi listrik menggunakan tiga algoritma yang diukur dengan metode pengukuran RMSE dengan hyperparameter yang sama. Perbandingan RMSE tiga algoritma yaitu RNN(0,623) , LSTM(0,602), dan GRU(0,589). Hasil yang didapatkan adalah GRU sedikit lebih baik daripada dua algoritma lainnya.
8	Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction	Wang, Jinjiang Yan, Jianxing Li, Chen Gao, Robert X. Zhao, Rui (2019)	GRU	Hasilnya adalah model heterogeneous GRU yang diusulkan dapat memberikan hasil yang dapat mengungguli beberapa model seperti SVR, CNN, LSTM dan BD-GRU dalam memprediksi performa mesin manufaktur. Dalam penelitian ini model menggunakan metode <i>hybrid prediction</i> . Pengukuran dilakukan dengan RMSE dan MAE. Pengukuran juga mencakupi rentang prediksi (prediksi 5 hari kedepan, 10 hari kedepan, 30 hari kedepan) dan menggunakan pembagian jumlah data training dan testing yang berbeda.
9	Gated Recurrent Unit – Recurrent Neural Network Untuk Peramalan Nilai Tukar Mata Uang Rupiah Terhadap Dolar Amerika	Tyas Nuur Khoolish (2017)	GRU	Hasilnya adalah model GRU yang dapat memprediksi nilai tukar mata uang dolar lebih baik ketimbang standart RNN. Hasil akhir setelah mendapatkan <i>hyper parameter</i> yang tepat pada GRU adalah GRU memiliki RMSE sebesar 62.82839, MAE sebesar 45.62825, MAPE sebesar 0.342715, dan D(stat) sebesar 69.67%. Sedangkan RNN mendapatkan hasil RMSE sebesar 63.803, MAE sebesar 46.91268, MAPE sebesar 0.352233, dan D(stats) sebesar 58.20%. dapat dilihat GRU lebih unggul daripada RNN.

Berdasarkan penelusuran terhadap penelitian terdahulu, terdapat beberapa penelitian yang memiliki karakteristik sama. Perbedaan penelitian ini dengan penelitian sebelumnya adalah objek dan metode yang digunakan. Beberapa penelitian telah melakukan penelitian terhadap data *time series* menggunakan algoritma GRU dengan studi kasus yang berbeda-beda. Penelitian ini mempunyai objek prediksi harga saham dengan metode GRU.

BAB III METODOLOGI PENELITIAN DAN PENGEMBANGAN SISTEM

3.1 Metodologi Penelitian

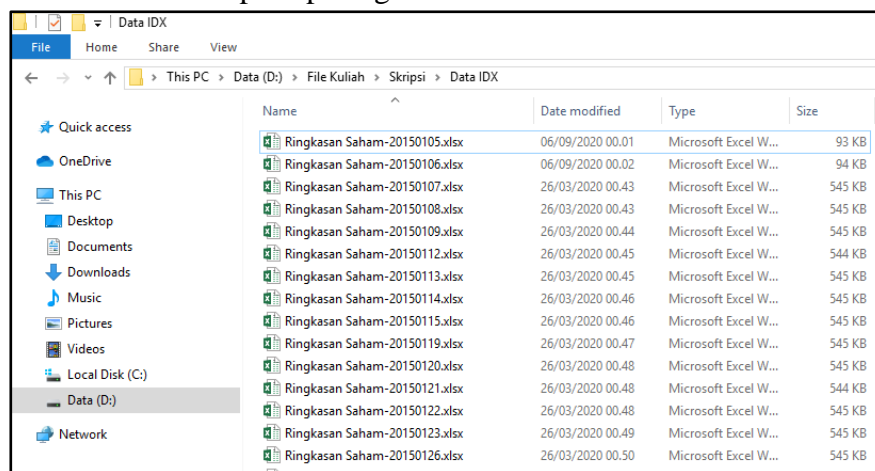
Metodologi penelitian pada penelitian ini adalah metode kuantitatif dengan data sekunder. Berikut adalah ilustrasi penelitian seperti pada gambar 3.1.



Gambar 3.1 Metodologi penelitian

3.1.1 Pengumpulan Data

Data yang digunakan dalam penelitian ini adalah ringkasan saham dari website Bursa Efek Indonesia (BEI) atau Stock Exchange Indonesia (IDX) (<https://www.idx.co.id/data-pasar/ringkasan-perdagangan/ringkasan-saham/>). Data yang diambil merupakan data dari tanggal 5 Januari 2015 sampai 31 Desember 2019. Data ringkasan saham BEI dapat didownload secara gratis atau *open source*. Data yang didownload berupa data-data ringkasan saham pada tanggal tertentu. Selenium WebDriver Python digunakan dalam proses download untuk memudahkan pengambilan data agar tidak dilakukan secara manual klik tetapi download otomatis. Hasil dari download menggunakan Selenium ini adalah data per hari dalam bentuk xlsx seperti pada gambar 3.2.



Gambar 3.2 Data IDX per hari

Dataset belum dapat digunakan karena bentuk dari dataset harus dalam bentuk satu perusahaan. Fitur yang terkandung dataset per hari adalah Kode Saham, Nama Perusahaan, Remarks, Sebelumnya, Open Price, First Trade, Tertinggi, Terendah, Penutupan, Selisih, Volume, Nilai, Frekuensi, Index Individual, Listed Shares, Offer, Offer Volume, Bid, Bid Volume, Last Trading Date, Tradeable Shares, Weight for Index, Foreign Sell, Foreign Buy, Non Regular Volume, Non Regular Value, dan Non Regular Frequency. Dataset ini mengandung banyak fitur yang tidak digunakan karena dataset hanya perlu lima fitur yaitu harga pembuka(*open*), harga tertinggi(*high*), harga terendah(*low*), harga penutupan(*close*), dan Volume, semua data ini dapat disingkat OHLCV. Dataset ini dilakukan *data cleaning* dan seleksi fitur dengan proses menghapus fitur selain OHLCV. Contoh Hasil *data cleaning* dan seleksi fitur seperti tabel 3.1.

Tabel 3.1 Dataset 31 Desember 2019

No	Kode Saham	Open Price	Tertinggi	Terendah	Penutupan	Volume
1	AALI	14500	14675	14300	14575	1202700
2	ABBA	107	110	106	106	21870500
3	ABDA	6975	0	0	6975	0
4	ABMM	1530	0	0	1530	0
....
673	ZONE	0	505	494	496	10000

Setelah dibersihkan dari fitur yang tidak diperlukan tahap selanjutnya adalah menggabungkan semua dataset atau disebut *data integration*. Tahap pertama *data integration* adalah menambah label dataset dengan tanggal. Tahap kedua adalah

menggabungkan semua dataset harian dengan tools Excel agar ditumpuk sesuai kolom. Keluaran *data integration* adalah satu file xlsl yang memuat semua data harian. Tahap selanjutnya agar dataset dapat dipakai adalah memisah dataset dengan masing-masing perusahaan. Proses pemisahan dataset ini menggunakan tools *google colab* yaitu notebook python. Setelah dipisahkan maka dataset akan siap pakai seperti contoh pada tabel 3.2 yang merupakan dataset dari perusahaan Adaro Energy (ADRO). Visualisasi data harga penutupan saham ADRO dari tahun 2015 sampai tahun 2019 pada gambar 3.3.

Tabel 3.2 Dataset Adaro Energy (ADRO)

Tanggal	Open Price	Tertinggi	Terendah	Penutupan	Volume
2015-01-05	1050.0	1050.0	1015.0	1025.0	23166900.0
2015-01-06	1005.0	1025.0	1005.0	1010.0	15700200.0
2015-01-07	1015.0	1030.0	1000.0	1010.0	18307300.0
2015-01-08	1010.0	1015.0	980.0	985.0	96054200.0
2015-01-09	995.0	1020.0	990.0	1005.0	52135900.0
2015-01-12	1010.0	1015.0	995.0	995.0	21558400.0
2015-01-13	990.0	1000.0	960.0	965.0	88452200.0
2015-01-14	955.0	970.0	945.0	945.0	78095500.0
2015-01-15	955.0	955.0	930.0	935.0	99177300.0
2015-01-19	950.0	955.0	940.0	945.0	37412900.0
2015-01-20	955.0	985.0	935.0	980.0	83960100.0
2015-01-21	985.0	1010.0	970.0	1005.0	82277900.0
2015-01-22	1015.0	1020.0	1000.0	1015.0	65512400.0
2015-01-23	1025.0	1030.0	985.0	1000.0	62477500.0
2015-01-26	990.0	1000.0	975.0	980.0	24910400.0
.....
2019-12-30	1565	1590	1555	1555	49275500



Gambar 3.3 Pergerakan harga penutupan ADRO

Penelitian ini akan menggunakan 6 perusahaan sebagai dataset percobaan. Dataset yang dipakai dipilih berdasarkan dua jenis saham berdasarkan tingkat likuiditas yang berbeda yaitu tinggi dan rendah. Likuiditas adalah kemampuan suatu perusahaan dalam melunasi hutang atau bisa diartikan kualitas kinerja suatu perusahaan. Dataset berasal dari tiga sektor yang berbeda yang terdaftar pada *Jakarta Stock Industrial Classification* (JASICA). Dari parameter sektor dan likuiditas maka dataset maka dataset berasal dari delapan saham yang masuk kedalam LQ45 (Liquid) dan delapan saham lainnya adalah saham Non-Liquid. Dasar pemilihan ini merujuk pada penelitian Faurina, 2019. Tabel 3.3

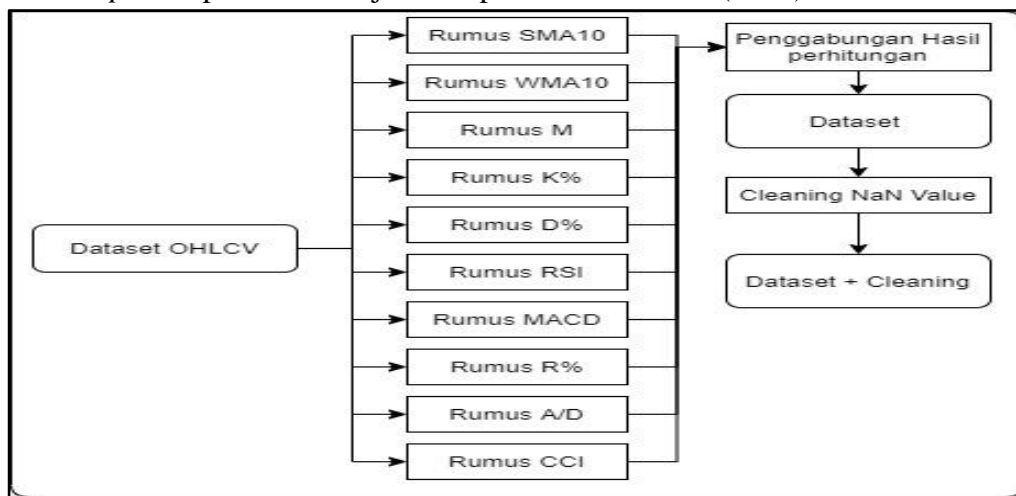
adalah daftar perusahaan yang akan diujikan dengan highlight biru adalah saham dengan Liquid dan yang tidak di-highlight adalah saham Non-Liquid.

Tabel 3.3 Daftar Dataset yang digunakan

Sektor	Saham	Quote
Mining	PT. Adaro Energy Tbk	ADRO
	PT. Vale Indonesia Tbk	INCO
Miscellaneous	PT. Astra International Tbk	ASII
	PT. Indomobil Sukses Internasional Tbk	IMAS
Infrastructure	PT. Telekomunikasi Indonesia (Persero) Tbk	TLKM
	PT. Garuda Indonesia (Persero) Tbk	GIAA

3.1.2 *Praprocessing data*

Proses ini berfungsi sebagai pengolah dataset agar dapat digunakan dan dilatih oleh RNN-GRU. Ada dua bentuk dataset yang akan diujikan yaitu data dengan bentuk *Technical Analysis* (TA) dan *Chartist*. TA adalah nilai yang biasa dipakai oleh para broker untuk analisis saham yang telah dijabarkan pada bab 2.4 . Implementasi dari TA menggunakan *library* python ta-lib (<https://github.com/mrjbq7/ta-lib>). Proses dari *praprocessing* TA dilakukan dengan *tools* google colab dengan ilustrasi proses pada gambar 3.4 . Pada tabel 3.4 adalah parameter yang harus ditetapkan untuk menghitung nilai *Technical Analysis*. Dasar pemilihan *time period* pada TA dirujuk dari penelitian Faurina (2019).



Gambar 3.4 Praprocessing TA
Tabel 3.4 Parameter Technical Analysis

No	Technical Analysis	Validasi
1	Simple 10-day Moving Average(SMA10)	Time period = 10
2	Weighted 10-day Moving Average(WMA10)	Time period = 10
3	Momentum (M)	Time period = 10
4	Stochastic K% (K%)	K period = 5
5	Stochastic D% (D%)	D period=3
6	Relative Strength Index (RSI)	Time Period =14
7	Moving Average Convergence Divergence(MACD)	Fast period = 12 Slow Period =26
8	Larry William's R% (R%)	Time Period = 14
9	Accumulation / Distribution Oscillator (A/D)	Time Period = 10
10	Commodity Channel Index (CCI)	Time Period = 14

Berikut beberapa contoh perhitungan pada *praprocessing* TA dengan contoh data ADRO pada tabel 3.2 dan validasi data pada tabel 3.4 :

1. *Simple 10-day Moving Average* (SMA10)

Contoh data yang diambil adalah data 10 hari dari tanggal 5 sampai tanggal 19.

$$\begin{aligned} \text{SMA10} &= \frac{C_t + C_{t-1} + \dots + C_{t-9}}{10} \\ \text{SMA10} &= \frac{945 + 935 + 945 + 965 + 995 + 1005 + 985 + 1010 + 1010 + 1025}{10} \\ \text{SMA10} &= \frac{9820}{10} \\ \text{SMA10} &= 982 \end{aligned}$$

2. *Weighted 10-day Moving Average* (WMA10)

Contoh data yang diambil adalah data 10 hari dari tanggal 5 sampai tanggal 19.

$$\begin{aligned} (n) \times C_t &= (10) \times 945 = 9450 \\ (n-1) \times C_{t-1} &= (10-1) \times 935 = 8415 \\ (n-2) \times C_{t-2} &= (10-2) \times 945 = 7560 \\ (n-3) \times C_{t-3} &= (10-3) \times 965 = 6755 \\ (n-4) \times C_{t-4} &= (10-4) \times 995 = 5970 \\ (n-5) \times C_{t-5} &= (10-5) \times 1005 = 5025 \\ (n-6) \times C_{t-6} &= (10-6) \times 985 = 3940 \\ (n-7) \times C_{t-7} &= (10-7) \times 1010 = 3030 \\ (n-8) \times C_{t-8} &= (10-8) \times 1010 = 2020 \\ (n-9) \times C_{t-9} &= (10-9) \times 1025 = 1025 \\ \text{WMA10} &= \frac{(n) \times C_t + (n-1) \times C_{t-1} + \dots + (n-9) \times C_{t-9}}{(n + (n-1) + (n-2) + \dots + 1)} \end{aligned}$$

WMA10

$$\begin{aligned} &= \frac{9450 + 8415 + 7560 + 6755 + 5970 + 5025 + 3940 + 3030 + 2020 + 1025}{(10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1)} \\ \text{WMA10} &= 967.09090909 \end{aligned}$$

3. *Momentum*(M)

Contoh data yang diambil adalah data 10 hari ($n = 10$) dari tanggal 5 sampai tanggal 19.

$$\begin{aligned} M &= C_t - C_{t-n} \\ M &= 945 - 1025 \\ M &= -80 \end{aligned}$$

4. *Stochastic K%* (K%)

Pada rumus dibutuhkan nilai tertinggi dan terendah dari suatu periode. Maka tahap pertama dilakukan pencarian nilai tertinggi dan terendah pada suatu periode. Contoh data yang diambil adalah data 5 hari ($n = 5$) dari tanggal 5 sampai tanggal 9.

$$\begin{aligned} LL_{t-n} &= 985 \\ HH_{t-n} &= 1025 \\ K\% &= \frac{C_t - LL_{t-n}}{HH_{t-n} - LL_{t-n}} \times 100 \end{aligned}$$

$$K\% = \frac{1005 - 985}{1025 - 985} \times 100$$

$$K\% = \frac{20}{40} \times 100$$

$$K\% = 50$$

5. *Stochastic D% (D%)*

Pada rumus ini dibutuhkan tiga nilai K%. Contoh data yang diambil adalah data 7 hari dari tanggal 5 sampai tanggal 13. Dengan periode $K\%_1$ dari tanggal 5 – tanggal 9, $K\%_2$ dari tanggal 6 – tanggal 12, $K\%_3$ dari tanggal 7 – tanggal 13.

$$K\%_1 = \frac{1005 - 985}{1025 - 985} \times 100 = 50$$

$$K\%_2 = \frac{995 - 985}{1010 - 985} \times 100 = 40$$

$$K\%_3 = \frac{1005 - 965}{1010 - 965} \times 100 = 89$$

$$D\% = \frac{\sum_{i=0}^{n-1} K_{t-i}\%}{n}$$

$$D\% = \frac{50 + 40 + 89}{3}$$

$$D\% = 59.67$$

6. *Relative Strength Index (RSI)*

Pada rumus ini dibutuhkan selisih harga penutupan setiap hari. Contoh data yang diambil adalah data dari tanggal 5 sampai tanggal 26 pada tabel 3.2 yaitu data 15 hari harga penutupan. Ilustrasi dari perhitungan selisih dapat dilihat pada gambar 3.5.

$$AVG\ gain = \frac{\sum_{i=0}^{n-1} Up_{t-i}/n}{14}$$

$$AVG\ gain = 7.142$$

$$AVG\ loss = \frac{\sum_{i=0}^{n-1} Dw_{t-i}/n}{14}$$

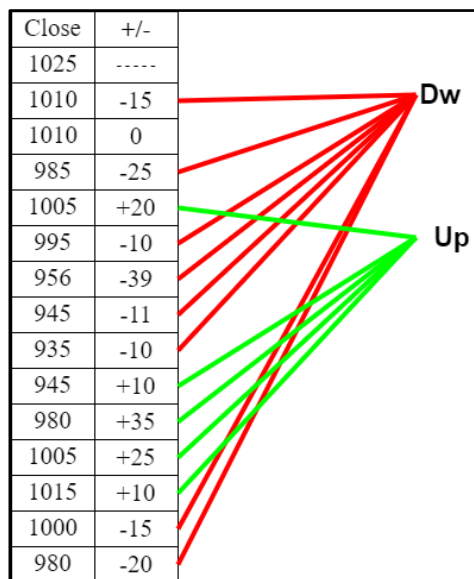
$$AVG\ loss = 10.357$$

$$RSI = 100 - \frac{100}{1 + \frac{(AVG\ gain)}{(AVG\ loss)}}$$

$$RSI = 100 - \frac{100}{1 + \frac{(7.142)}{(10.357)}}$$

$$RSI = 100 - \frac{100}{1 + 0.689}$$

$$RSI = 40.8$$



Gambar 3.5 Ilustrasi Selisih Harga Penutupan

7. *Moving Average Convergence Divergence*(MACD)

Pada tabel 3.4 periode waktu yang dipakai adalah 12 hari dan 26 hari. Namun pada contoh perhitungan dibawah ini menggunakan periode waktu 7 hari dan 14 hari. Langkah dasar dalam perhitungan MACD adalah hitung nilai EMA 7 *day* dan EMA 14 *day* (pada penelitian ini menggunakan EMA 12 *day* dan EMA 26 *day*) langkah dasar dalam menghitung EMA(k) pertama hitung SMA(k) *day* (dalam kasus ini 7 hari dan 14 hari). Kedua hitung nilai α menggunakan menggunakan persamaan 2.9. Ketiga hitung EMA(k) menggunakan persamaan 2.8. Setelah didapatkan nilai EMA 7 *day* dan EMA 14 *day* maka hitung MACD menggunakan EMA dengan *time step* yang sama dengan persamaan 2.7.

Perhitungan EMA 7 *day* : EMA(k) maka k = 7

Pertama hitung SMA(k) :

$$SMA(7) = \frac{1025 + 1010 + 1010 + 985 + 1005 + 995 + 965}{7} = 999.28571$$

Kedua Hitung α :

$$\alpha = \frac{2}{1+k} = \frac{2}{1+7} = \frac{2}{8} = 0.25$$

Ketiga hitung EMA :

Catatan untuk EMA yang pertama, nilai dari $EMA(k)_{t-1}$ adalah nilai SMA(k). Untuk nilai EMA yang kedua dan seterusnya menggunakan nilai EMA sebelumnya.

$$EMA(k)_t = EMA(k)_{t-1} + \alpha \times (C_t - EMA(k)_{t-1})$$

$$EMA(k)_t = 999.28571 + 0.25 \times (965 - 999.28571)$$

$$EMA(k)_t = 985.71429$$

Perhitungan EMA 14 *day* : EMA(k) maka k = 14

Pertama hitung SMA(k) :

$$\begin{aligned}
 &SMA(7) \\
 &= \frac{1025 + 1010 + 1010 + 985 + 1005 + 995 + 965 + 945 + 935 + 945 + 980 + 1005 + 1015 + 1000 + 980}{14} \\
 &= 987
 \end{aligned}$$

Kedua Hitung α :

$$\alpha = \frac{2}{1+k} = \frac{2}{1+14} = \frac{2}{15} = 0.13333$$

Ketiga hitung EMA :

Catatan untuk EMA yang pertama, nilai dari $EMA(k)_{t-1}$ adalah nilai SMA(k). Untuk nilai EMA yang kedua dan seterusnya menggunakan nilai EMA sebelumnya.

$$EMA(k)_t = EMA(k)_{t-1} + \alpha \times (C_t - EMA(k)_{t-1})$$

$$EMA(k)_t = 987 + 0.1333 \times (980 - 987)$$

$$EMA(k)_t = 986.19048$$

Setelah didapatkan nilai EMA maka tahap selanjutnya menggunakan persamaan 2.7 untuk menghitung nilai MACD.

$$MACD = EMA \text{ 7 day} - EMA \text{ 14 day}$$

$$MACD = 987.99268 - 986.19048$$

$$MACD = 1.8022$$

Ilustrasi perhitungan MACD ada pada tabel 3.5. *highlight* kuning merupakan SMA(k). Nilai akan terus dihitung dengan pada setiap nilai C_t .

Tabel 3.5 ilustrasi perhitungan MACD

Penutupan	EMA 7 hari	EMA 14 hari	MACD
1025			
1010			
1010			
985			
1005			
995			
965	999.28571		
945	985.71429		
935	973.03571		
945	966.02679		
980	969.52009		
1005	978.39007		
1015	987.54255		
1000	990.65691	987	
980	987.99268	986.19048	1.8022

8. *Larry William's R%* (R%)

Pada rumus dibutuhkan nilai tertinggi dan terendah dari suatu periode. Contoh data yang diambil adalah data dari tanggal 5 sampai 23.

$$H_n = 935$$

$$L_n = 1025$$

$$R\% = \frac{H_n - C_t}{H_n - L_n} \times 100$$

$$R\% = \frac{1025 - 1000}{1025 - 935} \times 100$$

$$R\% = \frac{25}{90} \times 100$$

$$R\% = 27.8$$

9. *Accumulation / Distribution Oscillator (A/D)*

Pada rumus ini contoh data yang diambil adalah data dari tanggal 5 sampai tanggal 19.

$$A/D = \frac{C_t - O_t}{(H_t - L_t) * V}$$

$$A/D = \frac{1025 - 1050}{(1050 - 1015) * 23166900}$$

$$A/D = 3.0832166335837521883130056860681e - 8$$

10. *Commodity Channel Index (CCI)*

Pada rumus ini SM_t merupakan rata-rata pergerakan M_t pada n periode. D_t adalah *mean deviation* dari pergerakan M_t . Contoh perhitungan dibawah menggunakan periode waktu 5 hari.

$$M_t = \frac{H_t + L_t + C_t}{3}$$

$$M_t = \frac{1050 + 1015 + 1025}{3}$$

$$M_t = \frac{3090}{3}$$

$$M_t = 1030$$

Untuk mencari nilai SM_t pada waktu periode 5 ($n = 5$) maka dibutuhkan 4 nilai lagi.

$$M_1 = 1030$$

$$M_2 = \frac{H_2 + L_2 + C_2}{3} = \frac{1025 + 1005 + 1010}{3} = 1013.3333$$

$$M_3 = \frac{H_3 + L_3 + C_3}{3} = \frac{1030 + 1000 + 1010}{3} = 1013.3333$$

$$M_4 = \frac{H_4 + L_4 + C_4}{3} = \frac{1015 + 980 + 985}{3} = 993.3333$$

$$M_5 = \frac{H_5 + L_5 + C_5}{3} = \frac{1020 + 990 + 1005}{3} = 1005$$

$$SM_t = \frac{\sum_{i=1}^n M_i}{n}$$

$$SM_t = \frac{1030 + 1013.333 + 1013.333 + 993.333 + 1005}{5}$$

$$SM_t = 1011$$

$$D_t = \frac{\sum_{i=1}^n |M_i - SM_t|}{n}$$

$$D_t = \frac{|1030 - 1011| + |1013.33 - 1011| + |1013.33 - 1011| + |993.33 - 1011| + |1005 - 1011|}{5}$$

$$D_t = 9.466666667$$

$$CCI = \frac{M_t - SM_t}{0.015 D_t}$$

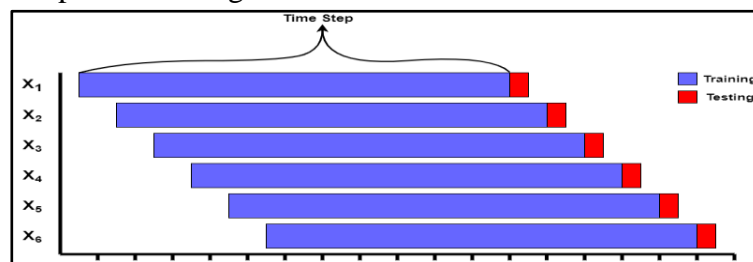
$$CCI = \frac{1005 - 1011}{0.015 * 9.46667}$$

$$CCI = \frac{-6}{0.142}$$

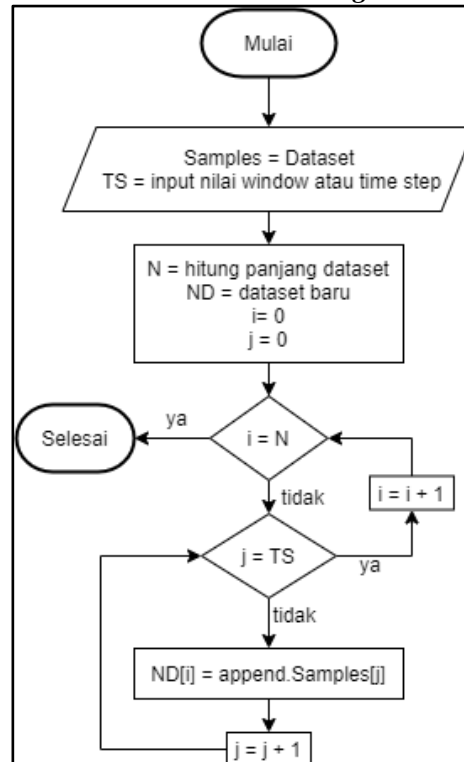
$$CCI = -42.253521126760563380281690140845$$

Implementasi dari *chartist* adalah data mentah dari *open, high, low, close* (OHLC). Data *chartist* dapat menjadi data latih karena memiliki serangkaian informasi mentah seperti yang telah dijabarkan pada bab 2.4. *Chartist* menggunakan metode *sliding window* yang telah dijabarkan pada bab 2.5.2 agar ketika memasuki jaringan ANN, ANN memiliki informasi yang cukup dalam melakukan *training* dalam *time step* tertentu. Ilustrasi *sliding window* terdapat pada gambar 3.6 dan flowchart *sliding window* terdapat pada gambar 3.7. Contoh dari *sliding window* ada pada tabel 3.7 yang merupakan *sliding window* dari tabel 3.6. *Highlight* pada tabel 3.6 dan 3.7 merupakan visualisasi perubahan *time step*. *Time step* yang dipakai adalah 26 hari karena merujuk pada *time period* TA yang paling panjang yang terdapat pada MACD. Penyamaan *time period* ini dilakukan agar kedua variasi data TA dan *chartist* mempunyai bobot informasi yang sama satu sama lain namun.

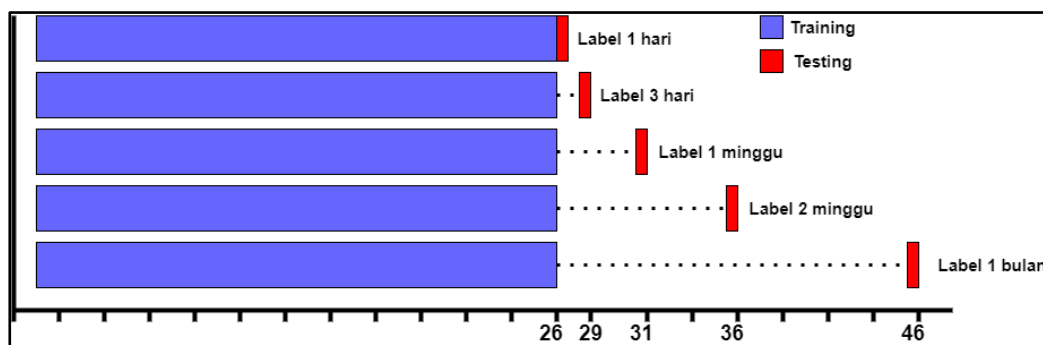
Penelitian ini menggunakan lima label berbeda berdasarkan panjang waktu yang ingin diprediksi. Lima label ini dapat menjadi alat ukur sejauh mana kemampuan RNN-GRU dapat memprediksi nilai di masa depan. Variasi label ini dapat menjadi hipotesis bahwa prediksi dengan rentang waktu pendek lebih akurat daripada rentang waktu yang panjang. Label yang digunakan adalah 1 hari, 3 hari, 1 minggu, 2 minggu, dan 1 bulan. Ilustrasi pelabelan dataset dapat dilihat di gambar 3.8.



Gambar 3.6 Ilustrasi Sliding Window



Gambar 3.7 Flowchart Sliding Window



Gambar 3.8 Pelabelan

Tabel 3.6 Contoh dataset

Open(O)	High(H)	Low(L)	Close(C)
14500	14675	14300	14575
14100	14525	14100	14500
13600	14200	13575	14100
13525	13625	13525	13600
13300	13525	13100	13525
13550	13550	13300	13300

Tabel 3.7 Contoh Sliding Window

O1	H1	L1	C1	O2	H2	L2	C2	O3	H3	L3	C3
14500	14675	14300	14575	14100	14525	14100	14500	13600	14200	13575	14100
14100	14525	14100	14500	13600	14200	13575	14100	13525	13625	13525	13600
13600	14200	13575	14100	13525	13625	13525	13600	13300	13525	13100	13525
13525	13625	13525	13600	13300	13525	13100	13525	13550	13550	13300	13300

Apabila semua *praprocessing* sudah dilakukan maka langkah selanjutnya adalah normalisasi dengan menggunakan metode *min max scaler*. Flowchart dari proses *min max scaler* terdapat pada gambar 3.7 . Contoh hasil tabel yang telah ternormalisasi ada pada tabel 3.8 yang merupakan hasil normalisasi *min max scaler* dari tabel 3.9. Tahapan *min max scaler* adalah sebagai berikut.

1. Inputkan dataset yang akan di normalisasi
2. Hitung berapa jumlah kolom dan jumlah index di dataset
3. Hitung nilai tertinggi ($\max(x)$) dan nilai terendah ($\min(x)$) pada kolom
4. Hitung menggunakan persamaan 2.17 pada setiap data
5. Muat nilai normalisasi ke dalam dataset baru
6. Ulang tahap 3 sampai 5 hingga setiap kolom atau fitur sudah ternormalisasi

Tabel 3.8 Contoh data normalisasi *min max scaler*

Open(O)	High(H)	Low(L)	Close(C)
1	1	1	1
0,666667	0,869565	0,833333	0,941176
0,25	0,586957	0,395833	0,627451
0,1875	0,086957	0,354167	0,235294
0	0	0	0,176471
0,208333	0,021739	0,166667	0

Berikut adalah contoh perhitungan *min max scaler* pada kolom *open* pada tabel 3.8 pada baris ke 2:

$$\min(x) = 13300$$

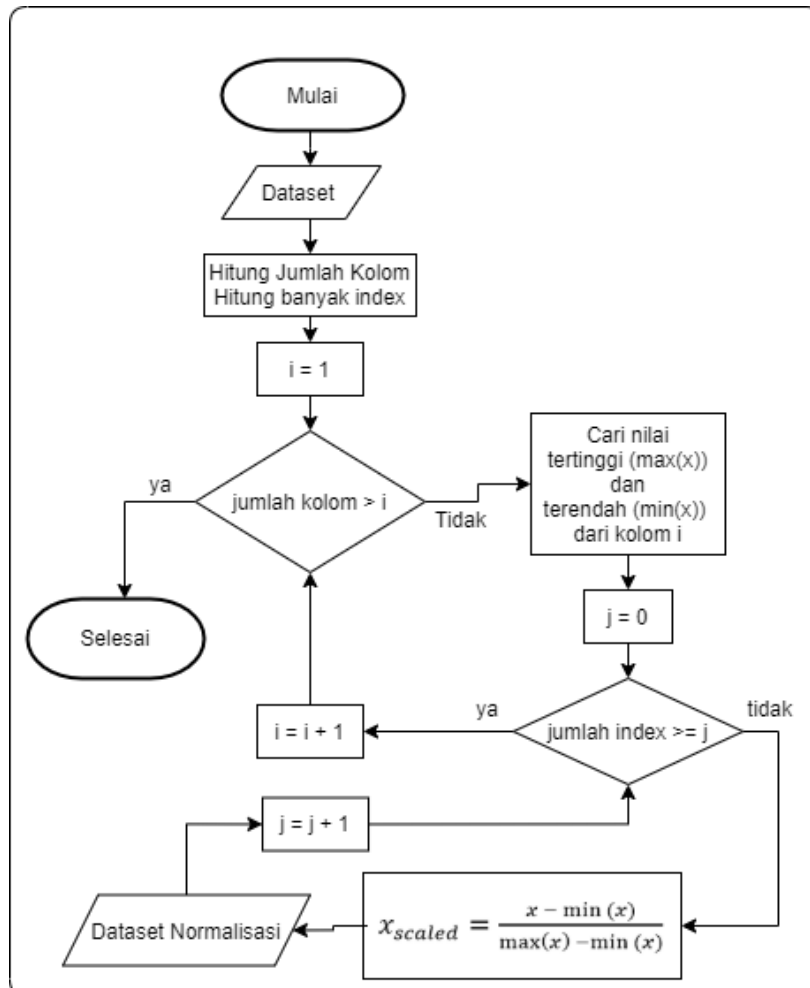
$$\max(x) = 14500$$

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

$$x_{scaled} = \frac{14100 - 13300}{14500 - 13300}$$

$$x_{scaled} = \frac{800}{1200}$$

$$x_{scaled} = 0,666667$$



Gambar 3.9 Flowchart Min Max Scaler

3.1.3 Proses Pra pengujian arsitektur

Proses sebelum melakukan proses uji arsitektur RNN-GRU yaitu pembagian dataset atau pembagian partisi. Dataset akan dibagi kedalam tiga bagian yaitu *training*, *validation*, dan *testing*. Panduan baku dalam menentukan jumlah partisi data ini belum mempunyai aturan baku. Menurut Khoolish, 2017 partisi yang paling sering dipakai adalah 70% data *training* dan 30% data *testing*. Pada penelitian ini akan menggunakan 70% data *training*, 10% data *validasi*, dan 20% data *testing*. Data *training* merupakan data yang akan dilatih didalam jaringan. Data *validation* adalah data uji dalam *training*, data ini akan dilatih ketika data *training* sudah dilatih terlebih dahulu. Data *testing* adalah data yang digunakan untuk menguji model.

3.1.4 Pengujian Arsitektur

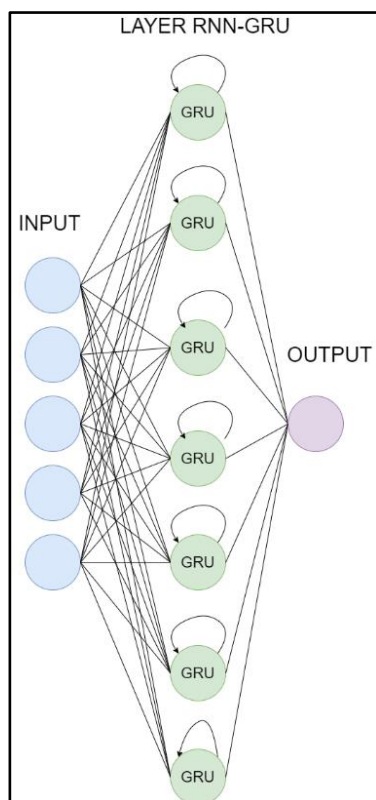
Pengujian arsitektur mempunyai tujuan untuk menemukan arsitektur terbaik yang dapat memprediksi harga saham dengan tepat. Pengujian arsitektur menggunakan dua parameter yaitu *Hidden layer (shallow dan stacked)* dan banyaknya unit GRU di *hidden layers*. Sebelum membandingkan dua parameter tersebut, jaringan ANN mempunyai parameter lainnya yang harus diinisialisasi terlebih dahulu diantaranya adalah unit layer output, fungsi aktivasi layer output, fungsi aktivasi input, fungsi aktivasi *reccurrent*, *epoch*, optimasi *gradient descent*, *batch*, *loss function*.

Unit didalam layer output sebanyak satu karena model ini akan memprediksi satu nilai. Fungsi aktivasi layer output yang digunakan adalah Sigmoid. Optimasi *gradient descent* yang digunakan adalah ADAM. ADAM merupakan adaptif *learning rate* yang mempunyai kemampuan dalam menemukan bobot dan bias dengan konvergensi tinggi seperti yang telah dijabarkan pada bab 2.8 . *Batch* yang digunakan adalah 128. *Batch* tidak mempunyai aturan baku dalam penentuan jumlahnya sampai sekarang. *Loss function* yang digunakan adalah MAE. MAE merupakan *loss function* yang digunakan pada data *time series* yang mempunyai toleransi terhadap outlier seperti data saham. *Epoch* maksimal yang digunakan adalah 300. Epoch tidak mempunyai aturan baku dalam penentuannya sampai sekarang. Untuk meminimalisir *Overfitting* karena epoch yang terlalu sering maka pada saat training model penelitian ini menggunakan *library tensorflow ModelCheckpoint* dengan *tools* “BestModel H5”. *Tools* ini berfungsi sebagai penyimpanan model terbaik selama epoch berlangsung. Berikut tabel 3.9 untuk parameter RNN-GRU yang diinisialisasi pertama kali.

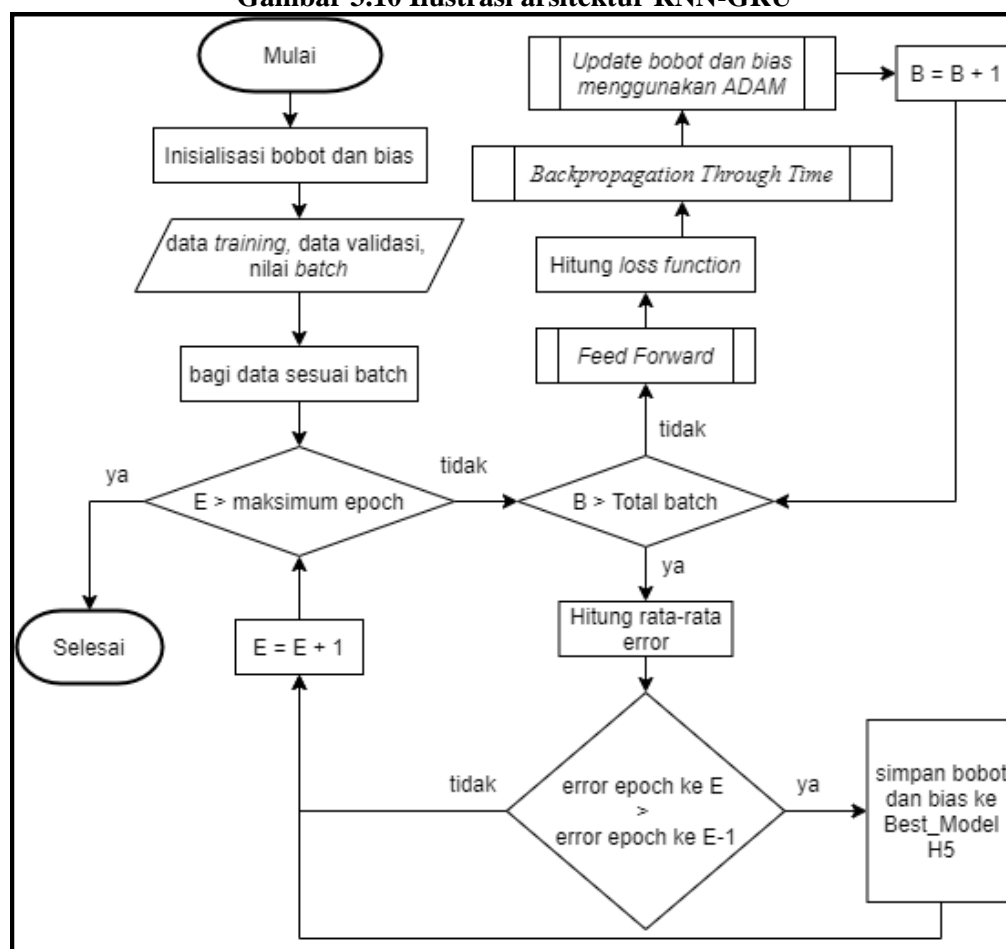
Tabel 3.9 Parameter RNN-GRU tetap

No	Parameter	Validasi
1	Unit layer output	1
2	Fungsi aktivasi output	Sigmoid
3	<i>Epoch</i>	300
4	Optimasi <i>gradient descent</i>	ADAM
5	<i>Batch</i>	128
6	<i>Loss function</i>	MAE

Arsitektur RNN-GRU tidak jauh beda dengan ANN pada umumnya. Perbedaan besar pada jaringan RNN-GRU dengan ANN biasa adalah layer *reccurrent* dan unit GRU nya. Pada layer ini unit atau biasa disebut dengan neuron akan melakukan pengiriman informasi dari unit sebelumnya kepada unit selanjutnya ketika melakukan *training*. Ilustrasi RNN-GRU ada pada gambar 3.10 dan ilustrasi unit GRU terdapat pada gambar 2.15. Untuk *training* RNN-GRU tidak berbeda jauh dengan ANN pada umumnya, namun parameter yang dilatih lebih banyak dan kompleks. *Backpropagation* pada RNN-GRU disebut dengan BPTT atau *Backpropagation Through Time*. BPTT melakukan pencarian gradient terhadap bobot dan bias terhadap waktu. BPTT mempunyai konsep yang sama dengan *Backpropagation* biasa namun dengan pencarian *gradient* yang lebih banyak dan kompleks. Ilustrasi Pelatihan jaringan RNN-GRU dapat dilihat pada gambar 3.11.



Gambar 3.10 Ilustrasi arsitektur RNN-GRU



Gambar 3.11 Flowchart Training RNN-GRU

Training RNN-GRU dimulai dengan inisialisasi bobot dan bias. Pada penelitian ini bobot dan bias diinisialisasi secara random dari nilai. Inisialisasi random pada bobot dan bias merupakan regulasi standar dalam jaringan ANN. Inisialisasi random akan menimbulkan *error* yang cukup tinggi pada awal training namun akan turun drastis karena algoritma *gradient descent*. Contoh dari inisialisasi ada pada tabel 3.11 dengan dengan contoh data input memakai data *technical analysis* pada tabel 3.10 yang mana memiliki sepuluh fitur. Sepuluh fitur ini artinya pada satu neuron bobot terhadap input (W_z, W_r, W_h) akan diinisialisasi sebanyak sepuluh. Contoh perhitungan pada unit GRU penelitian ini hanya satu unit atau neuron saja. Perhitungan pada neuron atau unit lain tidak berbeda.

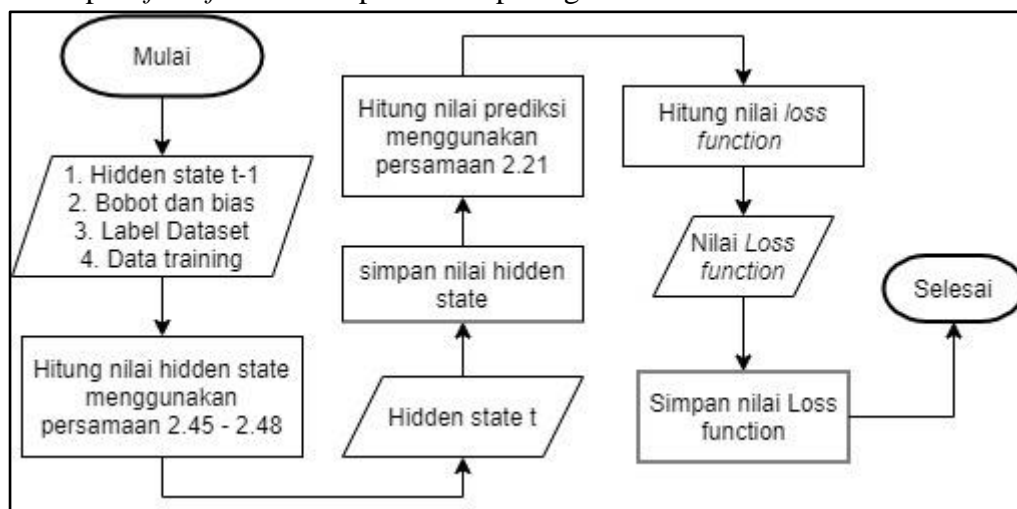
Tabel 3.10 Contoh dataset TA

N0	SMA10	WMA10	M	K%	D%	RSI	MACD	R%	A/D	CCI
1	0.9092	0.9035	0.3120	0.0217	0.1304	0.5495	0.6326	0.3730	0.4330	0.4450
2	0.9032	0.8938	0.3798	0.1428	0.0822	0.5098	0.6161	0.2136	0.4106	0.3224

Tabel 3.11 Contoh inisialisasi parameter RNN-GRU

Parameter	Nilai	Parameter	Nilai	Parameter	Nilai
W_z^1	-0.1367529	W_r^1	0.23016408	W_h^1	0.28899997
W_z^2	-0.1753298	W_r^2	-0.23431524	W_h^2	-0.334841
W_z^3	0.24726654	W_r^3	0.35121238	W_h^3	-0.20672792
W_z^4	0.03136447	W_r^4	-0.17205839	W_h^4	-0.28585976
W_z^5	-0.22678675	W_r^5	0.21602917	W_h^5	0.27622768
W_z^6	-0.06249793	W_r^6	0.19085133	W_h^6	0.3353057
W_z^7	0.07656327	W_r^7	-0.35337275	W_h^7	0.21361546
W_z^8	-0.1455175	W_r^8	0.04085408	W_h^8	-0.3104262
W_z^9	-0.17697811	W_r^9	0.28247118	W_h^9	-0.30179897
W_z^{10}	-0.1191807	W_r^{10}	-0.398562	W_h^{10}	-0.20790976
U_z	-0.0829422	U_r	-0.12283201	U_h	-0.01982183
b_z	1.0005621	b_r	0.0009534	b_h	-0.0009269

Setelah inisialisasi bobot maka nilai *batch* yang sebelumnya diinisialisasi akan dipakai dalam untuk membagi data sesuai nilai *batch*. Dataset akan dilatih berulang ulang sebanyak 300 *epoch*. *Feed forward* atau *forward phase* merupakan tahapan dimana jaringan akan berusaha memprediksi dengan nilai bobot dan bias yang telah diinisialisasi atau telah di *update*. Tahapan *feed forward* dapat dilihat pada gambar 3.12.



Gambar 3.12 Flowchart Feed forward RNN-GRU

Berikut adalah contoh salah satu perhitungan manual dan tahapan pada *feed forward* RNN-GRU dengan menggunakan contoh dataset pada tabel 3.10 dan inisialisasi parameter pada tabel 3.11. Langkah pertama adalah menghitung nilai z_t dengan persamaan 2.45. h_{t-1} diinisialisasi 0 karena perhitungan merupakan time step pertama. Input pada time step ke t (x_t) merupakan matriks dengan bentuk $A_{1 \times 10}$ dan bobot terhadap input (W_z, W_r, W_h) merupakan matriks dengan bentuk $B_{10 \times 1}$.

$$x_t = [0.9092 \quad 0.9035 \quad 0.3120 \quad 0.0217 \quad 0.1304 \quad 0.5495 \quad 0.6326 \quad 0.3730 \quad 0.4330 \quad 0.4450]$$

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

$$W_z = \begin{bmatrix} -0.1367529 \\ -0.1753298 \\ 0.24726654 \\ 0.03136447 \\ -0.22678675 \\ -0.06249793 \\ 0.07656327 \\ -0.1455175 \\ -0.17697811 \\ -0.1191807 \end{bmatrix}$$

$$U_z = -0.0829422$$

$$b_z = 1.0005621$$

$$W_z x_t = -0.40434508$$

$$U_z h_{t-1} = 0$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$z_t = \sigma(-0.40434508 + 0 + 1.0005621)$$

$$z_t = \sigma(0.59621702)$$

$$z_t = 0.6447903$$

Setelah didapatkan nilai z_t maka langkah selanjutnya mencari nilai r_t dengan persamaan 2.46.

$$W_r = \begin{bmatrix} 0.23016408 \\ -0.23431524 \\ 0.35121238 \\ -0.17205839 \\ 0.21602917 \\ 0.19085133 \\ -0.35337275 \\ -0.3104262 \\ 0.28247118 \\ -0.398562 \end{bmatrix}$$

$$U_r = -0.01982183$$

$$b_h = 0.0009534$$

$$W_r x_t = -0.15793368$$

$$U_r h_{t-1} = 0$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

$$r_t = \sigma(-0.15793368 + 0 + 0.0009534)$$

$$r_t = \sigma(-0.1583834)$$

$$r_t = 0.4604867$$

Setelah didapatkan nilai r_t maka langkah selanjutnya mencari nilai \tilde{h}_t dengan persamaan 2.47.

$$W_h = \begin{bmatrix} 0.28899997 \\ -0.334841 \\ -0.20672792 \\ -0.28585976 \\ 0.27622768 \\ 0.3353057 \\ 0.21361546 \\ -0.3104262 \\ -0.30179897 \\ -0.20790976 \end{bmatrix}$$

$$U_h = -0.12283201$$

$$b_h = -0.0009269$$

$$W_h x_t = -0.0940564$$

$$U_h(r_t \odot h_{t-1}) = 0$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$

$$\tilde{h}_t = \tanh(-0.0940564 + 0 + -0.0009269)$$

$$\tilde{h}_t = \tanh(-0.0949833)$$

$$\tilde{h}_t = -0.0946987$$

Setelah didapatkan nilai \tilde{h}_t maka langkah selanjutnya mencari nilai h_t dengan persamaan 2.48.

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

$$h_t = (1 - 0.6447903) \odot 0 + 0.6447903 \odot -0.0946987$$

$$h_t = 0 + -0.061060$$

$$h_t = -0.061060$$

Pada time step kedua jaringan akan menggunakan data input yang kedua dan menggunakan nilai h_t sebelumnya sebagai h_{t-1} . Bobot dan bias masih menggunakan hasil inisialisasi pada tabel 3.11, bobot dan bias akan berubah pada saat akhir *epoch* yang akan diupdate oleh BPTT dan ADAM. Berikut adalah contoh perhitungan manual pada time step kedua. Perhitungan pada time step selanjutnya sama dengan time step kedua.

$$x_t = [0.9032 \quad 0.8938 \quad 0.3798 \quad 0.1428 \quad 0.0822 \quad 0.5098 \quad 0.6161 \quad 0.2136 \quad 0.4106 \quad 0.3224]$$

$$h_{t-1} = -0.061060$$

$$W_z = \begin{bmatrix} -0.1367529 \\ -0.1753298 \\ 0.24726654 \\ 0.03136447 \\ -0.22678675 \\ -0.06249793 \\ 0.07656327 \\ -0.1455175 \\ -0.17697811 \\ -0.1191807 \end{bmatrix}$$

$$U_z = -0.0829422$$

$$b_z = 1.0005621$$

$$W_z x_t = -0.32728801$$

$$U_z h_{t-1} = 0.005064$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$z_t = \sigma(-0.32728801 + 0.0050644 + 1.0005621)$$

$$z_t = \sigma(0.67833809)$$

$$z_t = 0.663368$$

Setelah didapatkan nilai z_t maka langkah selanjutnya mencari nilai r_t dengan persamaan 2.46.

$$W_r = \begin{bmatrix} 0.23016408 \\ -0.23431524 \\ 0.35121238 \\ -0.17205839 \\ 0.21602917 \\ 0.19085133 \\ -0.35337275 \\ -0.3104262 \\ 0.28247118 \\ -0.398562 \end{bmatrix}$$

$$U_r = -0.01982183$$

$$b_h = 0.0009534$$

$$W_r x_t = -0.07413605$$

$$U_r h_{t-1} = 0.0003929$$

$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

$$r_t = \sigma(-0.07413605 + 0.0003929 + 0.0009534)$$

$$r_t = \sigma(-0.07278975)$$

$$r_t = 0.48181059$$

Setelah didapatkan nilai r_t maka langkah selanjutnya mencari nilai \tilde{h}_t dengan persamaan 2.47.

$$W_h = \begin{bmatrix} 0.28899997 \\ -0.334841 \\ -0.20672792 \\ -0.28585976 \\ 0.27622768 \\ 0.3353057 \\ 0.21361546 \\ -0.3104262 \\ -0.30179897 \\ -0.20790976 \end{bmatrix}$$

$$U_h = -0.12283201$$

$$b_h = -0.0009269$$

$$W_h x_t = -0.08949425$$

$$U_h(r_t \odot h_{t-1}) = 0.00361364$$

$$\tilde{h}_t = \tanh(W_h x_t + U_h(r_t \odot h_{t-1}) + b_h)$$

$$\tilde{h}_t = \tanh(-0.08949425 + 0.00361364 + -0.0009269)$$

$$\tilde{h}_t = \tanh(-0.08680751)$$

$$\tilde{h}_t = -0.086590118$$

Setelah didapatkan nilai \tilde{h}_t maka langkah selanjutnya mencari nilai h_t dengan persamaan 2.48.

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

$$h_t = (1 - 0.663368) \odot (-0.061060) + 0.663368 \odot (-0.086590118)$$

$$h_t = -0.02055475 + -0.0574411$$

$$h_t = -0.03688635$$

Proses diatas akan terus berlangsung sampai pada *time step* terakhir pada setiap neuron. *Feed forward* akan dilanjutkan dengan perhitungan di layer output. Input pada layer output sendiri adalah bobot, bias, dan h_t . Layer output pada penelitian ini adalah satu *neuron* dengan fungsi aktivasi Sigmoid. Berikut adalah contoh perhitungan di layer output dengan

anggapan ada 3 neuron pada layer GRU pada arsitektur dengan menggunakan persamaan 2.21. nilai x_i pada persamaan ini adalah nilai dari h_t . Nilai x_i dan W_i pada perhitungan ini adalah nilai random hanya sebagai contoh perhitungan.

$$Y_t = \sigma\left(\sum_{i=0}^n x_i W_i + b_t\right)$$

$$Y_t = \sigma((x_1 W_1 + x_2 W_2 + x_3 W_3) + b_t)$$

$$Y_t = \sigma((-0.04635 * 0.3156 + (-0.0710) * 0.1889 + 0.0757 * 0.2272) + 0.0087)$$

$$Y_t = \sigma((-0.01462 + -0.01341 + 0.0172) + 0.0087)$$

$$Y_t = \sigma(-0.01083 + 0.0087)$$

$$Y_t = \sigma(-0.00213)$$

$$Y_t = 0.4995$$

Setelah melakukan *feed forward* langkah selanjutnya adalah menghitung *loss function* untuk mengukur *error* pada dataset untuk setiap *epoch*. Berikut adalah contoh perhitungan *loss function* menggunakan persamaan MAE dengan 3 *time step* dan label yang ternormalisasi.

$$MAE = \frac{\sum_{i=1}^N |y_i - \hat{y}_i|}{N}$$

$$MAE = \frac{\sum_{i=1}^3 |y_i - \hat{y}_i|}{3}$$

$$MAE = \frac{|y_1 - \hat{y}_1| + |y_2 - \hat{y}_2| + |y_3 - \hat{y}_3|}{3}$$

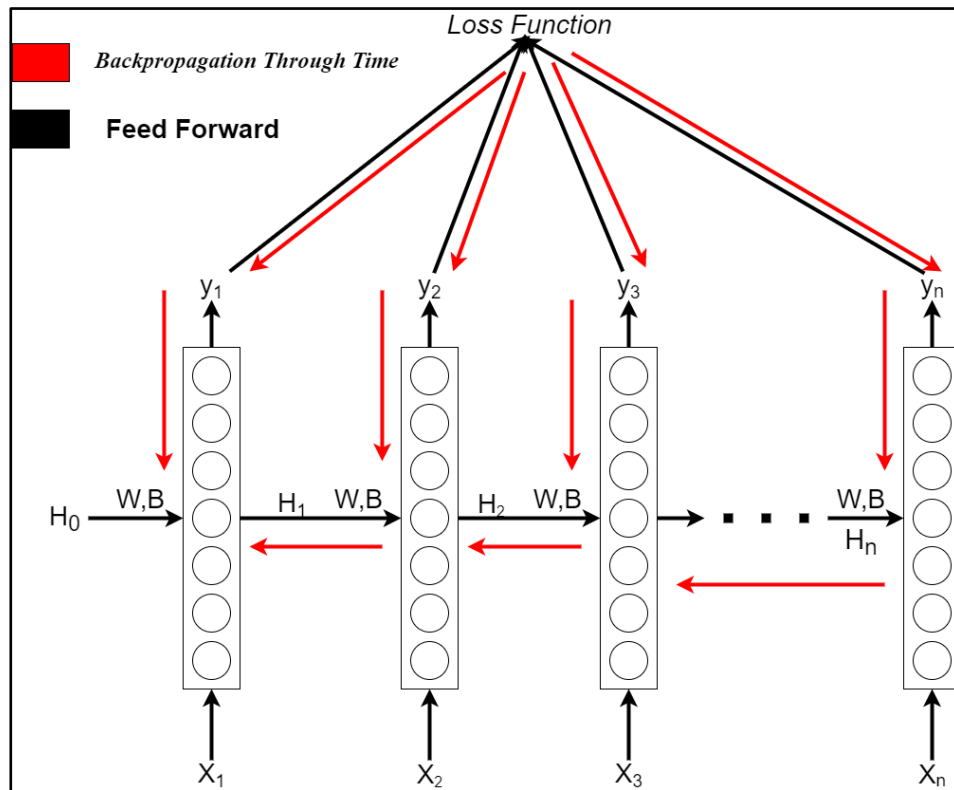
$$MAE = \frac{|0.4995 - 0.4102| + |0.6731 - 0.5991| + |0.5001 - 0.5219|}{3}$$

$$MAE = \frac{0.0893 + 0.074 + 0.0218}{3}$$

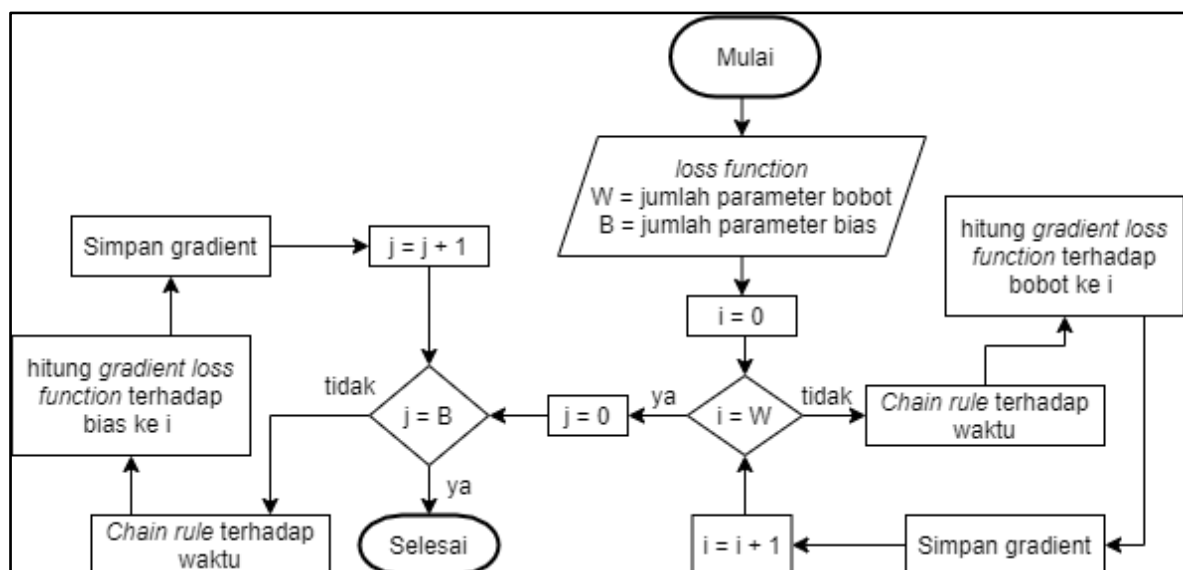
$$MAE = 0.1851$$

Nilai *loss function* berfungsi untuk proses selanjutnya yaitu *Backpropagation Through Time*(BPTT). Proses BPTT akan mengeluarkan output *gradient*. *Gradient* pada proses ini berfungsi sebagai ukuran seberapa sensitif nilai *error* atau *loss function* terhadap parameter RNN-GRU. Nilai *gradient* akan dihitung dengan aturan *chain rule* yang telah dijabarkan pada bab 2.8 . Namun pada *Backpropagation Through Time* (BPTT) ada sedikit perbedaan dengan *Backpropagation* (BP) pada umumnya. Beda BPTT dan BP adalah ketika BP mengukur kesensitifan *loss function* terhadap parameter tidak mengukur bobot sebelumnya, namun pada BPTT harus mengukur *gradient* parameter sebelumnya karena beberapa parameter dipengaruhi data sebelumnya. Berikut ilustrasi BPTT pada gambar 3.13 dan flowchart BPTT pada gambar 3.14.

Tahapan dari BPTT yang pertama adalah mencari persamaan untuk gradient terhadap parameter yang ingin dicari dengan metode turunan *chain rule*. Setelah mendapatkan persamaan gradient maka gradient tersebut dihitung. Lakukan proses ini terhadap seluruh parameter di RNN-GRU.

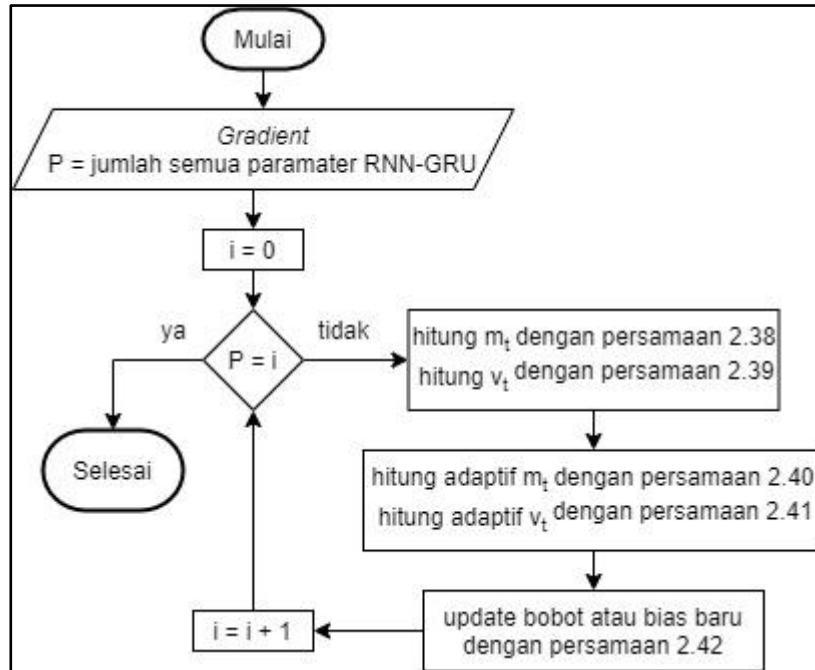


Gambar 3.13 Backpropagation Through Time



Gambar 3.14 Flowchart Backpropagation Through Time

Setelah mendapatkan *gradient* pada setiap parameter di dalam jaringan maka langkah selanjutnya dalam training RNN-GRU adalah melakukan *update* terhadap bobot dan bias. Proses ini menggunakan metode ADAM. Penjelasan dan perhitungan ADAM dapat dilihat pada bab 2.8. *flowchart* ADAM dapat dilihat pada gambar 3.15.



Gambar 3.15 Flowchart ADAM

Sebelum melakukan Pengujian nilai error dengan MAE, RMSE, dan DA, data testing dan hasil prediksi harus melewati proses Denormalisasi. Denormalisasi adalah proses mengembalikan nilai-nilai yang sudah di *praprocessing* kembali menjadi nilai-nilai sebelum di *praprocessing*. Pada penelitian ini *praprocessing* yang merubah nilai dataset adalah *Min Max Scaler*. Proses Denormalisasi untuk *praprocessing Min Max Scaler* adalah membalik rumus *Min Max Scaler*. Berikut persamaan dalam denormalisasi *Min Max Scaler*

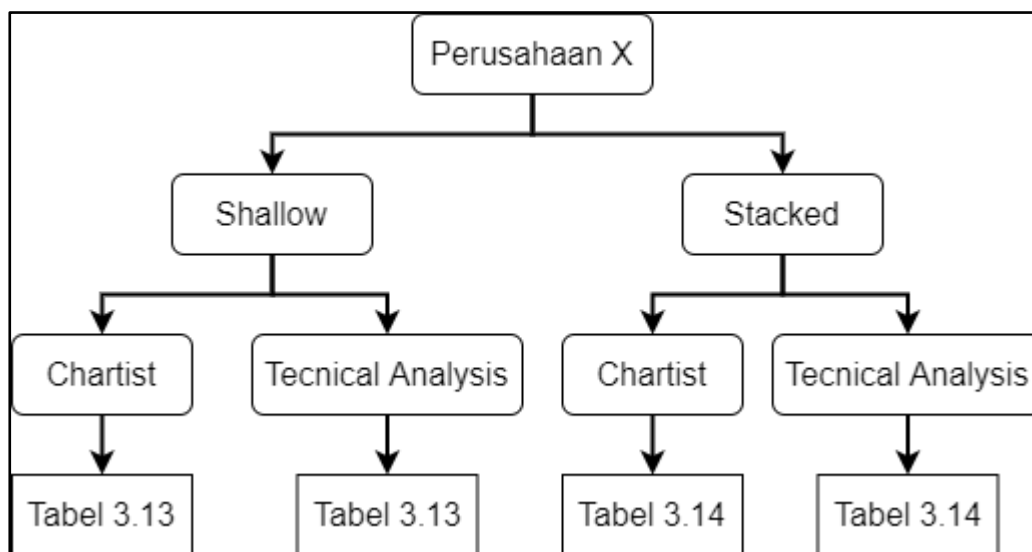
$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

$$x_{scaled} \times (\max(x) - \min(x)) = x - \min(x)$$

$$x_{scaled} \times (\max(x) - \min(x)) + \min(x) = x$$

$$x = x_{scaled} \times (\max(x) - \min(x)) + \min(x)$$

Nilai prediksi dari model dan nilai aktual pada data *test* yang sudah dinormalisasi akan di denormalisasi dengan persamaan diatas agar mendapatkan nilai error yang relevan. Untuk pengujian arsitektur penelitian ini menguji dua arsitektur *hidden layer* RNN yaitu *Shallow* dan *Stacked*. *Shallow* adalah arsitektur RNN-GRU satu *layer*. *Stacked* adalah arsitektur RNN-GRU lebih dari satu *layer* dengan arah *time step* yang sama dengan *layer* sebelumnya. Pada penelitian ini arsitektur *stacked* mempunyai 2 *layer*. Pengujian dilakukan dengan dua bentuk data yaitu *chartist* dan *technical analysis*. Dari kombinasi arsitektur *hidden layer* dan bentuk data maka alur pengujian satu perusahaan dapat dilihat pada gambar 3.16.



Gambar 3.16 Alur Pengujian satu perusahaan
Tabel 3.13 Pengujian *Shallow*

Perusahaan	Label	Unit	MAE	RMSE	DA
Perusahaan X	1 hari	50			
		100			
		150			
		200			
	3 hari	50			
		100			
		150			
		200			
	1 minggu	50			
		100			
		150			
		200			
	2 minggu	50			
		100			
		150			
		200			
	1 bulan	50			
		100			
		150			
		200			

Tabel 3.14 Pengujian *Stacked*

Perusahaan	Label	Hidden layer 1	Hidden layer 2	MAE	RMSE	DA
Perusahaan X	1 hari	50	50			
		50	100			
		50	150			
		50	200			
		100	50			
		100	100			
		100	150			
		100	200			
		150	50			
		150	100			
		150	150			
		150	200			

Tabel 3.15 Pengujian *Stacked*(lanjutan)

Perusahaan	Label	Hidden layer 1	Hidden layer 2	MAE	RMSE	DA
		200	50			
		200	100			
		200	150			
		200	200			
	3 hari	50	50			
		50	100			
		50	150			
		50	200			
		100	50			
		100	100			
		100	150			
		100	200			
		150	50			
		150	100			
		150	150			
		150	200			
		200	50			
		200	100			
		200	150			
		200	200			
	1 minggu	50	50			
		50	100			
		50	150			
		50	200			
		100	50			
		100	100			
		100	150			
		100	200			
		150	50			
		150	100			
		150	150			
		150	200			
		200	50			
		200	100			
		200	150			
		200	200			
	2 minggu	50	50			
		50	100			
		50	150			
		50	200			
		100	50			
		100	100			
		100	150			
		100	150			
		150	50			
		150	100			
		150	150			
		150	200			
		200	50			
		200	100			
		200	150			
		200	200			
	1 bulan	50	50			

Tabel 3.16 Pengujian *Stacked*(lanjutan)

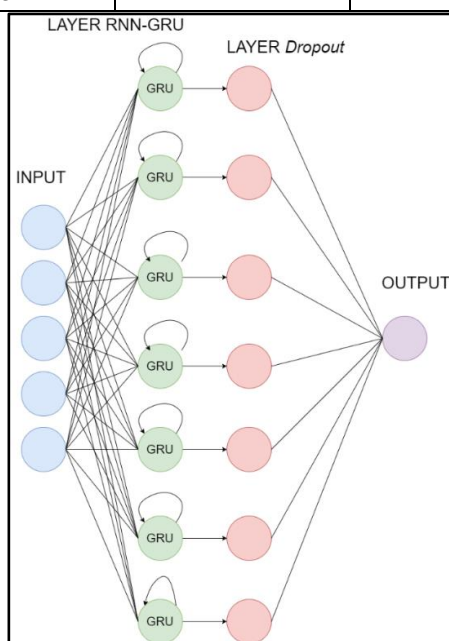
Perusahaan	Label	Hidden layer 1	Hidden layer 2	MAE	RMSE	DA
		50	100			
		50	150			
		50	200			
		100	50			
		100	100			
		100	150			
		100	200			
		150	50			
		150	100			
		150	150			
		150	200			
		200	50			
		200	100			
		200	150			
		200	200			

3.1.5 Pengujian *Dropout*

Dropout merupakan regulasi untuk menghindari *overfitting* sebagaimana telah dijelaskan pada bab 2.12 . Metode ini diimplementasikan dengan menambahkan layer diantara 2 layer yang mana layer ini akan mematikan neuronnya agar jaringan tersebut terputus. Layer ini akan berada antara layer GRU dan layer output. Berapa banyak neuron yang mati akan ditentukan dengan parameter yang diberikan. Parameter yang akan diujikan adalah 0.2 dan 0.5 artinya adalah 20% atau 50 % dari jaringan. Untuk penentuan parameter *dropout* belum ada ketentuan khusus sampai sekarang. Tabel pengujian dropout ada pada tabel 3.16.

Tabel 3.16 Pengujian Regulasi *Dropout*

Nama Model	<i>Dropout</i>	MAE	RMSE	DA
Model X	0.2			
	0.5			

**Gambar 3.17 Ilustrasi layer *Dropout***

DAFTAR PUSTAKA

- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agarwal, M. (2017). Back Propagation in Convolutional Neural Networks — Intuition and Code. medium. [online] Diakses di: <<https://becominghuman.ai/back-propagation-in-convolutional-neural-networks-intuition-and-code-714ef1c38199>> [Diakses 21 Desember 2020]
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>
- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>
- Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU). *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–7. <https://doi.org/10.1109/NCG.2018.8593076>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. *ArXiv*, 1–5.
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature*. (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- Dewi, P. D. A., & Suaryana, I. G. N. A. (2013). Pengaruh Eps, Der, Dan Pbv Terhadap Harga Saham. *E-Jurnal Akuntansi*, 4(1), 215–229.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397. <https://doi.org/10.1016/j.eswa.2011.02.068>
- Id, T. A. R., Abbas, D. K., & Turel, Y. K. (2019). A multi hidden recurrent neural network with a modified grey wolf optimizer. *PLOS ONE*, 1–23. Retrieved from <https://doi.org/10.1371/journal.pone.0213237>
- Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38(5), 5311–5319. <https://doi.org/10.1016/j.eswa.2010.10.027>
- Karlik, B., & Olgac, A. V. (2019). Performance Analysis of Various Activation Functions in

- Artificial Neural Networks. *Journal of Physics: Conference Series*, 1237(2), 111–122. <https://doi.org/10.1088/1742-6596/1237/2/022030>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111(December 2019), 352–372. <https://doi.org/10.1016/j.trc.2019.12.022>
- Raha, M. (2008). Go public. *Print Professional*, 46(2), 20–24. <https://doi.org/10.12968/nuwa.2018.9.37>
- Rasyidin, M. (2014). *Pengaruh Foreign Direct Investment terhadap Pengembangan Pasar Saham di Indonesia*. 53–68. Retrieved from papers3://publication/uuid/F63F59D4-38D1-4868-9789-FAF13B3E5D90
- Ratnayaka, R. M. K. T., Seneviratne, D. M. K. N., Jianguo, W., & Arumawadu, H. I. (2015). A hybrid statistical approach for stock market forecasting based on Artificial Neural Network and ARIMA time series models. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 54–60. <https://doi.org/10.1109/BESC.2015.7365958>
- Struye, J., & Latré, S. (2020). Hierarchical temporal memory and recurrent neural networks for time series prediction: An empirical validation and reduction to multilayer perceptrons. *Neurocomputing*, 396(xxxx), 291–301. <https://doi.org/10.1016/j.neucom.2018.09.098>
- Sugiartawan, P., Pulungan, R., & Sari, A. K. (2017). *Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network*. (June). <https://doi.org/10.14569/IJACSA.2017.080243>
- Wang, J., Yan, J., Li, C., Gao, R. X., & Zhao, R. (2019). Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111, 1–14. <https://doi.org/10.1016/j.compind.2019.06.001>
- Zhang, D., & Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: A GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7), 578–585. <https://doi.org/10.1049/iet-its.2017.0313>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Armano, G., Marchesi, M., & Murru, A. (2005). A hybrid genetic-neural architecture for stock indexes forecasting. *Information Sciences*, 170(1), 3–33. <https://doi.org/10.1016/j.ins.2003.03.023>
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature*. (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chandra, A. L. (2019). Learning Parameters, Part 5: AdaGrad, RMSProp, and Adam. towardsdatascience. [online] Diakses di: <<https://towardsdatascience.com/learning-parameters-part-5-65a2f3583f7d>> [Diakses 2 Februari 2021]
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- David, F. R. (2010). *Manajemen Strategis*, Jakarta. Gramedia.

- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>
- Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU). *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–7. <https://doi.org/10.1109/NCG.2018.8593076>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. *ArXiv*, 1–5.
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature*. (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- Dewi, P. D. A., & Suaryana, I. G. N. A. (2013). Pengaruh Eps, Der, Dan Pbv Terhadap Harga Saham. *E-Jurnal Akuntansi*, 4(1), 215–229.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397. <https://doi.org/10.1016/j.eswa.2011.02.068>
- Id, T. A. R., Abbas, D. K., & Turel, Y. K. (2019). A multi hidden recurrent neural network with a modified grey wolf optimizer. *PLOS ONE*, 1–23. Retrieved from <https://doi.org/10.1371/journal.pone.0213237>
- Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38(5), 5311–5319. <https://doi.org/10.1016/j.eswa.2010.10.027>
- Karlik, B., & Olgac, A. V. (2019). Performance Analysis of Various Activation Functions in Artificial Neural Networks. *Journal of Physics: Conference Series*, 1237(2), 111–122. <https://doi.org/10.1088/1742-6596/1237/2/022030>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111(December 2019), 352–372. <https://doi.org/10.1016/j.trc.2019.12.022>
- Raha, M. (2008). Go public. *Print Professional*, 46(2), 20–24. <https://doi.org/10.12968/nuwa.2018.9.37>
- Rasyidin, M. (2014). *Pengaruh Foreign Direct Investment terhadap Pengembangan Pasar Saham di Indonesia*. 53–68. Retrieved from [papers3://publication/uuid/F63F59D4-38D1-4868-9789-](https://publication.uuid/F63F59D4-38D1-4868-9789-)

- Ratnayaka, R. M. K. T., Seneviratne, D. M. K. N., Jianguo, W., & Arumawadu, H. I. (2015). A hybrid statistical approach for stock market forecasting based on Artificial Neural Network and ARIMA time series models. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 54–60. <https://doi.org/10.1109/BESC.2015.7365958>
- Struye, J., & Latré, S. (2020). Hierarchical temporal memory and recurrent neural networks for time series prediction: An empirical validation and reduction to multilayer perceptrons. *Neurocomputing*, 396(xxxx), 291–301. <https://doi.org/10.1016/j.neucom.2018.09.098>
- Sugiartawan, P., Pulungan, R., & Sari, A. K. (2017). *Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network*. (June). <https://doi.org/10.14569/IJACSA.2017.080243>
- Wang, J., Yan, J., Li, C., Gao, R. X., & Zhao, R. (2019). Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111, 1–14. <https://doi.org/10.1016/j.compind.2019.06.001>
- Zhang, D., & Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: A GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7), 578–585. <https://doi.org/10.1049/iet-its.2017.0313>
- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>
- Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU). *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–7. <https://doi.org/10.1109/NCC.2018.8593076>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. *ArXiv*, 1–5.
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature*. (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- Dewi, P. D. A., & Suaryana, I. G. N. A. (2013). Pengaruh Eps, Der, Dan Pbv Terhadap Harga Saham. *E-Jurnal Akuntansi*, 4(1), 215–229.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397. <https://doi.org/10.1016/j.eswa.2011.02.068>

- Id, T. A. R., Abbas, D. K., & Turel, Y. K. (2019). A multi hidden recurrent neural network with a modified grey wolf optimizer. *PLOS ONE*, 1–23. Retrieved from <https://doi.org/10.1371/journal.pone.0213237>
- Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38(5), 5311–5319. <https://doi.org/10.1016/j.eswa.2010.10.027>
- Karlik, B., & Olgac, A. V. (2019). Performance Analysis of Various Activation Functions in Artificial Neural Networks. *Journal of Physics: Conference Series*, 1237(2), 111–122. <https://doi.org/10.1088/1742-6596/1237/2/022030>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111(December 2019), 352–372. <https://doi.org/10.1016/j.trc.2019.12.022>
- Raha, M. (2008). Go public. *Print Professional*, 46(2), 20–24. <https://doi.org/10.12968/nuwa.2018.9.37>
- Rasyidin, M. (2014). *Pengaruh Foreign Direct Investment terhadap Pengembangan Pasar Saham di Indonesia*. 53–68. Retrieved from papers3://publication/uuid/F63F59D4-38D1-4868-9789-FAF13B3E5D90
- Ratnayaka, R. M. K. T., Seneviratne, D. M. K. N., Jianguo, W., & Arumawadu, H. I. (2015). A hybrid statistical approach for stock market forecasting based on Artificial Neural Network and ARIMA time series models. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 54–60. <https://doi.org/10.1109/BESC.2015.7365958>
- Struye, J., & Latré, S. (2020). Hierarchical temporal memory and recurrent neural networks for time series prediction: An empirical validation and reduction to multilayer perceptrons. *Neurocomputing*, 396(xxxx), 291–301. <https://doi.org/10.1016/j.neucom.2018.09.098>
- Sugiartawan, P., Pulungan, R., & Sari, A. K. (2017). *Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network*. (June). <https://doi.org/10.14569/IJACSA.2017.080243>
- Wang, J., Yan, J., Li, C., Gao, R. X., & Zhao, R. (2019). Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111, 1–14. <https://doi.org/10.1016/j.compind.2019.06.001>
- Zhang, D., & Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: A GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7), 578–585. <https://doi.org/10.1049/iet-its.2017.0313>
- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>

- Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU). *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–7. <https://doi.org/10.1109/NCG.2018.8593076>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. *ArXiv*, 1–5.
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature.* (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- Dewi, P. D. A., & Suaryana, I. G. N. A. (2013). Pengaruh Eps, Der, Dan Pbv Terhadap Harga Saham. *E-Jurnal Akuntansi*, 4(1), 215–229.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397. <https://doi.org/10.1016/j.eswa.2011.02.068>
- Id, T. A. R., Abbas, D. K., & Turel, Y. K. (2019). A multi hidden recurrent neural network with a modified grey wolf optimizer. *PLOS ONE*, 1–23. Retrieved from <https://doi.org/10.1371/journal.pone.0213237>
- Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38(5), 5311–5319. <https://doi.org/10.1016/j.eswa.2010.10.027>
- Karlik, B., & Olgac, A. V. (2019). Performance Analysis of Various Activation Functions in Artificial Neural Networks. *Journal of Physics: Conference Series*, 1237(2), 111–122. <https://doi.org/10.1088/1742-6596/1237/2/022030>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111(December 2019), 352–372. <https://doi.org/10.1016/j.trc.2019.12.022>
- Raha, M. (2008). Go public. *Print Professional*, 46(2), 20–24. <https://doi.org/10.12968/nuwa.2018.9.37>
- Rasyidin, M. (2014). *Pengaruh Foreign Direct Investment terhadap Pengembangan Pasar Saham di Indonesia*. 53–68. Retrieved from papers3://publication/uuid/F63F59D4-38D1-4868-9789-FAF13B3E5D90
- Ratnayaka, R. M. K. T., Seneviratne, D. M. K. N., Jianguo, W., & Arumawadu, H. I. (2015). A hybrid statistical approach for stock market forecasting based on Artificial Neural Network and ARIMA time series models. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 54–60. <https://doi.org/10.1109/BESC.2015.7365958>

- Struye, J., & Latré, S. (2020). Hierarchical temporal memory and recurrent neural networks for time series prediction: An empirical validation and reduction to multilayer perceptrons. *Neurocomputing*, 396(xxxx), 291–301. <https://doi.org/10.1016/j.neucom.2018.09.098>
- Sugiartawan, P., Pulungan, R., & Sari, A. K. (2017). *Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network*. (June). <https://doi.org/10.14569/IJACSA.2017.080243>
- Wang, J., Yan, J., Li, C., Gao, R. X., & Zhao, R. (2019). Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111, 1–14. <https://doi.org/10.1016/j.compind.2019.06.001>
- Zhang, D., & Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: A GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7), 578–585. <https://doi.org/10.1049/iet-its.2017.0313>
- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>
- Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU). *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–7. <https://doi.org/10.1109/NCG.2018.8593076>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. *ArXiv*, 1–5.
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature*. (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- Dewi, P. D. A., & Suaryana, I. G. N. A. (2013). Pengaruh Eps, Der, Dan Pbv Terhadap Harga Saham. *E-Jurnal Akuntansi*, 4(1), 215–229.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397. <https://doi.org/10.1016/j.eswa.2011.02.068>
- Id, T. A. R., Abbas, D. K., & Turel, Y. K. (2019). A multi hidden recurrent neural network with a modified grey wolf optimizer. *PLOS ONE*, 1–23. Retrieved from <https://doi.org/10.1371/journal.pone.0213237>
- Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38(5), 5311–5319. <https://doi.org/10.1016/j.eswa.2010.10.027>

- Karlik, B., & Olgac, A. V. (2019). Performance Analysis of Various Activation Functions in Artificial Neural Networks. *Journal of Physics: Conference Series*, 1237(2), 111–122. <https://doi.org/10.1088/1742-6596/1237/2/022030>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111(December 2019), 352–372. <https://doi.org/10.1016/j.trc.2019.12.022>
- Raha, M. (2008). Go public. *Print Professional*, 46(2), 20–24. <https://doi.org/10.12968/nuwa.2018.9.37>
- Rasyidin, M. (2014). *Pengaruh Foreign Direct Investment terhadap Pengembangan Pasar Saham di Indonesia*. 53–68. Retrieved from papers3://publication/uuid/F63F59D4-38D1-4868-9789-FAF13B3E5D90
- Ratnayaka, R. M. K. T., Seneviratne, D. M. K. N., Jianguo, W., & Arumawadu, H. I. (2015). A hybrid statistical approach for stock market forecasting based on Artificial Neural Network and ARIMA time series models. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 54–60. <https://doi.org/10.1109/BESC.2015.7365958>
- Struye, J., & Latré, S. (2020). Hierarchical temporal memory and recurrent neural networks for time series prediction: An empirical validation and reduction to multilayer perceptrons. *Neurocomputing*, 396(xxxx), 291–301. <https://doi.org/10.1016/j.neucom.2018.09.098>
- Sugiartawan, P., Pulungan, R., & Sari, A. K. (2017). *Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network*. (June). <https://doi.org/10.14569/IJACSA.2017.080243>
- Wang, J., Yan, J., Li, C., Gao, R. X., & Zhao, R. (2019). Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111, 1–14. <https://doi.org/10.1016/j.compind.2019.06.001>
- Zhang, D., & Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: A GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7), 578–585. <https://doi.org/10.1049/iet-its.2017.0313>

- Han, J., Pei, J., & Kamber, M. (2012). *Data Mining Concepts and Techniques Third Edition*, Waltham, USA. Elsevier.
- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>
- Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU). *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–7. <https://doi.org/10.1109/NCG.2018.8593076>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. *ArXiv*, 1–5.
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature*. (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- Dewi, P. D. A., & Suaryana, I. G. N. A. (2013). Pengaruh Eps, Der, Dan Pbv Terhadap Harga Saham. *E-Jurnal Akuntansi*, 4(1), 215–229.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397. <https://doi.org/10.1016/j.eswa.2011.02.068>
- Id, T. A. R., Abbas, D. K., & Turel, Y. K. (2019). A multi hidden recurrent neural network with a modified grey wolf optimizer. *PLOS ONE*, 1–23. Retrieved from <https://doi.org/10.1371/journal.pone.0213237>
- Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38(5), 5311–5319. <https://doi.org/10.1016/j.eswa.2010.10.027>
- Karlik, B., & Olgac, A. V. (2019). Performance Analysis of Various Activation Functions in Artificial Neural Networks. *Journal of Physics: Conference Series*, 1237(2), 111–122. <https://doi.org/10.1088/1742-6596/1237/2/022030>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111(December 2019), 352–372. <https://doi.org/10.1016/j.trc.2019.12.022>
- Raha, M. (2008). Go public. *Print Professional*, 46(2), 20–24. <https://doi.org/10.12968/nuwa.2018.9.37>
- Rasyidin, M. (2014). *Pengaruh Foreign Direct Investment terhadap Pengembangan Pasar Saham*

- di Indonesia*. 53–68. Retrieved from papers3://publication/uuid/F63F59D4-38D1-4868-9789-FAF13B3E5D90
- Ratnayaka, R. M. K. T., Seneviratne, D. M. K. N., Jianguo, W., & Arumawadu, H. I. (2015). A hybrid statistical approach for stock market forecasting based on Artificial Neural Network and ARIMA time series models. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 54–60. <https://doi.org/10.1109/BESC.2015.7365958>
- Struye, J., & Latré, S. (2020). Hierarchical temporal memory and recurrent neural networks for time series prediction: An empirical validation and reduction to multilayer perceptrons. *Neurocomputing*, 396(xxxx), 291–301. <https://doi.org/10.1016/j.neucom.2018.09.098>
- Sugiartawan, P., Pulungan, R., & Sari, A. K. (2017). *Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network*. (June). <https://doi.org/10.14569/IJACSA.2017.080243>
- Wang, J., Yan, J., Li, C., Gao, R. X., & Zhao, R. (2019). Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111, 1–14. <https://doi.org/10.1016/j.compind.2019.06.001>
- Zhang, D., & Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: A GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7), 578–585. <https://doi.org/10.1049/iet-its.2017.0313>
- Agarap, A. F. (2018). *Deep Learning using Rectified Linear Units (ReLU)*. (1), 2–8. Retrieved from <http://arxiv.org/abs/1803.08375>
- Agustin, M., & Prahasto, T. (2012). Penggunaan Jaringan Syaraf Tiruan Backpropagation Untuk Seleksi Penerimaan Mahasiswa Baru Pada Jurusan Teknik Komputer Di Politeknik Negeri Sriwijaya. *Jurnal Sistem Informasi Bisnis*, 2(2), 89–97. <https://doi.org/10.21456/vol2iss2pp089-097>
- Althelaya, K. A., El-Alfy, E. S. M., & Mohammed, S. (2018). Stock Market Forecast Using Multivariate Analysis with Bidirectional and Stacked (LSTM, GRU). *21st Saudi Computer Society National Computer Conference, NCC 2018*, 1–7. <https://doi.org/10.1109/NCG.2018.8593076>
- Arista, D. (2012). *ANALISIS FAKTOR – FAKTOR YANG MEMPENGARUHI RETURN SAHAM (Kasus pada Perusahaan Manufaktur yang Go Public di BEI periode. 3*, 1–15.
- Bock, S., Goppold, J., & Weiß, M. (2018). An improvement of the convergence proof of the ADAM-Optimizer. *ArXiv*, 1–5.
- Chai, T., & Oceanic, N. (2015). *Root mean square error (RMSE) or mean absolute error (MAE)? – Arguments against avoiding RMSE in the literature*. (February), 3–7. <https://doi.org/10.5194/gmd-7-1247-2014>
- Chen, J. X., Jiang, D. M., & Zhang, Y. N. (2019). A Hierarchical Bidirectional GRU Model With Attention for EEG-Based Emotion Classification. *IEEE Access*, 7, 118530–118540. <https://doi.org/10.1109/access.2019.2936817>
- Dewi, P. D. A., & Suaryana, I. G. N. A. (2013). Pengaruh Eps, Der, Dan Pbv Terhadap Harga Saham. *E-Jurnal Akuntansi*, 4(1), 215–229.
- Guresen, E., Kayakutlu, G., & Daim, T. U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, 38(8), 10389–10397. <https://doi.org/10.1016/j.eswa.2011.02.068>

- Id, T. A. R., Abbas, D. K., & Turel, Y. K. (2019). A multi hidden recurrent neural network with a modified grey wolf optimizer. *PLOS ONE*, 1–23. Retrieved from <https://doi.org/10.1371/journal.pone.0213237>
- Kara, Y., Acar Boyacioglu, M., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul Stock Exchange. *Expert Systems with Applications*, 38(5), 5311–5319. <https://doi.org/10.1016/j.eswa.2010.10.027>
- Karlik, B., & Olgac, A. V. (2019). Performance Analysis of Various Activation Functions in Artificial Neural Networks. *Journal of Physics: Conference Series*, 1237(2), 111–122. <https://doi.org/10.1088/1742-6596/1237/2/022030>
- Lecun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, 521(7553), 436–444. <https://doi.org/10.1038/nature14539>
- Ma, T., Antoniou, C., & Toledo, T. (2020). Hybrid machine learning algorithm and statistical time series model for network-wide traffic forecast. *Transportation Research Part C: Emerging Technologies*, 111(December 2019), 352–372. <https://doi.org/10.1016/j.trc.2019.12.022>
- Raha, M. (2008). Go public. *Print Professional*, 46(2), 20–24. <https://doi.org/10.12968/nuwa.2018.9.37>
- Rasyidin, M. (2014). *Pengaruh Foreign Direct Investment terhadap Pengembangan Pasar Saham di Indonesia*. 53–68. Retrieved from papers3://publication/uuid/F63F59D4-38D1-4868-9789-FAF13B3E5D90
- Ratnayaka, R. M. K. T., Seneviratne, D. M. K. N., Jianguo, W., & Arumawadu, H. I. (2015). A hybrid statistical approach for stock market forecasting based on Artificial Neural Network and ARIMA time series models. *2015 International Conference on Behavioral, Economic and Socio-Cultural Computing, BESC 2015*, (Besc), 54–60. <https://doi.org/10.1109/BESC.2015.7365958>
- Struye, J., & Latré, S. (2020). Hierarchical temporal memory and recurrent neural networks for time series prediction: An empirical validation and reduction to multilayer perceptrons. *Neurocomputing*, 396(xxxx), 291–301. <https://doi.org/10.1016/j.neucom.2018.09.098>
- Sugiartawan, P., Pulungan, R., & Sari, A. K. (2017). *Prediction by a Hybrid of Wavelet Transform and Long-Short-Term-Memory Neural Network*. (June). <https://doi.org/10.14569/IJACSA.2017.080243>
- Wang, J., Yan, J., Li, C., Gao, R. X., & Zhao, R. (2019). Deep heterogeneous GRU model for predictive analytics in smart manufacturing: Application to tool wear prediction. *Computers in Industry*, 111, 1–14. <https://doi.org/10.1016/j.compind.2019.06.001>
- Zhang, D., & Kabuka, M. R. (2018). Combining weather condition data to predict traffic flow: A GRU-based deep learning approach. *IET Intelligent Transport Systems*, 12(7), 578–585. <https://doi.org/10.1049/iet-its.2017.0313>