**PROYEKSI CURAH HUJAN DAERAH PADANG PARIAMAN  
MENGGUNAKAN *DEEP LEARNING* DENGAN  
METODE *LONG SHORT-TERM MEMORY***

**SKRIPSI**

*Untuk Memenuhi Salah Satu Syarat*

*Memperoleh Gelar Sarjana Komputer*

**Program Studi : Teknik Informatika  
Jenjang Pendidikan : Strata 1 (S1)**



**OLEH :**

**EDO SULAIMAN**  
**NIM. 18101152630092**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PUTRA INDONESIA “YPTK” PADANG**

**2021**

# BAB I PENDAHULUAN

## Latar Belakang

Dewasa ini data merupakan penunjang pengambilan keputusan secara cepat, atau dikenal dengan istilah *Data Driven Decision Making (DDDM)* di mana kemajuan teknologi berperan besar dalam memanfaatkan data dan informasi tersebut (Aditya, Mulyana, Eka, & Widianto, 2020). Dalam berbagai aspek termasuk melakukan prediksi mengenai informasi curah hujan yang akurat di mana pemodelan tersebut masih memiliki kekurangan seperti penggunaan jumlah *parameter*, asumsi matematis, dan rumusan persamaan yang cenderung rumit, untuk menghasilkan sebuah model prediksi yang mendekati keakuratan optimal harus memiliki banyak *paremeter* dan *variabel* *input* untuk memenuhi sebuah asumsi prediksi (Supriyadi, 2019).

Mengatasi perihal tersebut, dikembangkanlah sebuah *Kecerdasan Buatan (Artificial Inteligence)* yang memiliki kemampuan untuk melakukan pembelajaran untuk menganalisis berbagai macam asumsi dan aspek yang berpengaruh untuk menarik kesimpulan (Supriyadi, 2019)*. Artificial Intelligence (AI)* atau *kecerdasan buatan* didefinisikan secara berbeda dalam konteks yang berbeda pula dalam disiplin ilmu komputer AI adalah mempelajari cara menstimulasikan untuk melakukan tugas yang *bias*anya membutuhkan pemahaman seperti manusia (Knowledge@Wharton, 2018). AI juga suatu cabang ilmu komputer yang menggunakan lebih banyak simbol daripada angka, dan memproses informasi berdasarkan jumlah aturan dalam merepresentasikan pengetahuan (Swarnkar & Swarnkar, 2019).

*Machine Learning* adalah bagian dari AI di mana mesin digunakan untuk belajar dari pengalaman masa lalu (Aditya, Mulyana, Eka, & Widianto, 2020). Algoritma *Machine Learning* digunakan dalam berbagai aplikasi, seperti dalam kedokteran, pengenalan email, pengenalan suara, dan visi komputer, di mana sulit atau tidak mungkin untuk mengembangkan algoritma konvensional untuk melakukan tugas yang diperlukan (Hu, Niu, Carrasco, Lennox, & Arvin, 2020).

Beberapa implementasi *Machine Learning* menggunakan data dan *Neural Network (Jaringan Saraf)* dengan cara yang meniru kerja otak biologis manusia (Zhou, 2019). *Deep Learning* dapat dipahami sebagai bentuk *Neural Network layer* berganda yang merupakan bagian dari *Machine Learning* yang dapat digunakan dalam tugas termasuk *computer vision, speech recognition, natural language processing, machine translation, bioinformatics, drug design, medical image analysis, material inspection* dan *board game programs*, di mana mereka telah menghasilkan hasil yang sebanding dan dalam beberapa kasus melebihi kinerja para pakar (Hu, Niu, Carrasco, Lennox, & Arvin, 2020).

Misalnya, sebuah komputasi yang menggunakan *Deep Learning*, mampu memahami konsep seperti garis, bentuk, tekstur, dan juga pengaruhnya dengan melihat data-data citra tanpa bantuan tambahan dari manusia (Schneiderman & Kanade, 2002). *Machine Learning* senantiasa bekerja menggunakan 1 *layer* di mana *Deep Learning* bekerja lebih dari 1 *layer*. untuk batasan *layer* dari *Deep Learning* itu sendiri sebagai *Neural Network* *bias*anya memiliki 3 *layer* atau lebih, makin bayak *layer* yang digunakan akan memengaruhi lama waktu yang terpakai untuk komputer mengalkulasi (Hinton, et al., 2012).

*Layer* pada *Deep Learning* dapat di gambarkan seperti lapisan *neuron* pada otak manusia *layer* itu nantinya akan menggambarkan jarak atau vektor menggunakan Fungsi matematika yaitu fungsi *sigmoid (σ)* (Putra J. G., 2020)*.* Alasan fungsi *sigmoid* digunakan karena dalam fungsi ini membutuhkan perhitungan yang relatif mudah dan cepat. Selain itu, fungsi *sigmoid* dapat diartikan sebagai nilai peluang karena nilainya antara 0 dan 1 (Putra J. G., 2020).

Salah satu pendekatan *Deep Learning* yang mampu secara otomatis mempelajari fitur yang dideskripsikan dalam bentuk *vektor* adalah *Recurrent Neural Networks (RNN)* (Puspaningrum, Bunga, & Iryanto, 2020). Pada *RNN* sendiri teknik learning bekerja dengan menyimpan *layer* dari *output* kembali sebagai *input* pada *hidden* *layer* berikutnya hingga memprediksi hasil akhir (Tarkus, Sompie, & Jacobus, 2020). Kelemahan *RNN* adalah tidak mampu lagi untuk belajar menghubungkan informasi ketika ada kesenjangan yang terus tumbuh, memori yang tersimpan akan semakin tidak relevan seiring waktu berjalan karena tertimpa dengan memori baru (Putra, Osmond, & Ansori, 2020), di sebabkan kelemahan dari *RNN* sendiri tidak dapat mempelajari informasi yang terlalu jauh atau *Long-Term Dependencies*, yang cukup jauh pada masukannya (Wibisono & Khodra, 2018).

Penelitian *Deep Learning* terdahulu yang di lakukan Juanda, Jondri, & Rohmawati, tentang *Prediksi Harga Bitcoin Dengan Menggunakan Recurrent Neural Network* menurutnya, masalah prediksi *time series* adalah jenis pemodelan prediktif yang sulit, tidak seperti pemodelan predikftif regresi, *time series* juga menambah kompleksitas ketergantungan urutan antar variabel *input*. *Recurrent Neural Network* terbukti berhasil digunakan untuk prediksi data *time series* karena *RNN* mampu menggunakan informasi yang telah direkam sebelumnya yang panjang urutannya atau *sequence*-nya beragam-ragam. Oleh karena itu, pembangunan sistem ini dibuat dengan metode *Recurrent Neural Network* dengan menggunakan algoritma *Backpropagation Through Time*. Hasil akhir Prediksi harga Bitcoin dapat dilakukan menggunakan Recurrent Neural Network. Akurasi rata-rata terbaik yang didapatkan adalah 98.76% pada data latih dan 97.46% pada data uji, dengan *parameter* jumlah pola *input* terbaik adalah 5, jumlah *epoch* 1000, nilai *learning rate* 0.001 dan jumlah *hidden* unit 50 (Juanda, Jondri, & Rohmawati, 2018).

*Long Short-Term Memory (LSTM)* merupakan sebuah pengembangan metode dari arsitektur *Recurrent Neural Network (RNN)*, Banyak peneliti yang mengembangkan metode *LSTM* di berbagai bidang seperti dalam bidang prediksi deret waktu atau *forecasting* dikarenakan metode *LSTM* mampu mengatasi kekurangan tersebut karena metode ini dapat mengatur memori pada setiap masukannya dengan menggunakan memory cells dan gate *units* pada setiap *neuron*s yang berfungsi sebagai pengatur memori (Putra, Osmond, & Ansori, 2020). Contoh penggunaan *Deep Learning* untuk data *time series* yang banyak dihasilkan dari pengamatan cuaca adalah *LSTM*, *LSTM* sendiri diciptakan oleh Hochreiter dan Schmidhuber pada tahun 1997 (Supriyadi, 2019).

Penelitian *LSTM* terdahulu yang yang di teliti oleh Poornima & Pushpalatha, mengenai *Prediction of Rainfall Using Intensified LSTM Based Recurrent Neural Network with Weighted Linear Units* dalam penelitian tersebut menyajikan *Long Short-Term Memory (LSTM)* berbasis *Recurrent Neural Network (RNN)* untuk memprediksi *curah hujan*. *Neural Network* dilatih dan diuji menggunakan kumpulan data standar curah hujan. Jaringan yang dilatih akan menghasilkan atribut prediksi curah hujan. *Parameter* yang dipertimbangkan untuk evaluasi kinerja dan efisiensi model prediksi curah hujan yang diusulkan adalah *Root Mean Square Error (RMSE), akurasi, jumlah epoch, loss*, dan *learning rate*. Menurutnya *LSTM* dapat menyimpan data besar ke dalam memorinya dan dapat menghindari gradient yang hilang lebih baik dari pada *RNN* dan menunjukkan akurasi lebih baik di bandingkan *RNN*, *LSTM* pun juga mempertahankan akurasi di masa depan seiring dengan mempertimbangkan nilai *Root Mean Square Error (RMSE), akurasi, jumlah epoch, loss, dan learning rate* (Poornima & Pushpalatha, 2019)*.*

Penelitian *LSTM* terdahulu yang juga di lakukan oleh Supriyadi, mengenai metode *Deep Learning LSTM* untuk memprediksi *parameter* cuaca, seperti suhu udara, kelembaban, kecepatan angin, dan tekanan udara. Metode ini bekerja dengan memanfaatkan fungsi matematika seperti fungsi *tanh* dan *sigmoid* yang berada dalam *layer* *LSTM*. Adapun jumlah *layer* yang digunakan sebanyak 200 buah. Sedangkan jumlah datanya dibagi dua menjadi training data dan test data dengan rasio 9:1.pada bulan Januari 2019. Diperoleh *RMSE* *parameter* suhu udara, kelembaban, kecepatan angin, dan tekanan udara nilainya semakin baik ketika menggunakan *Deep Learning* *LSTM* dengan update dibandingkan *LSTM* tanpa update. Diperoleh hasil prediksi suhu udara, kelembaban, kecepatan angin, dan tekanan udara 1 hari ke depan memiliki *RMSE* yang baik. Dari *parameter* cuaca tersebut hanya *parameter* suhu dan kelembaban udara yang mengalami pertambahan *RMSE* seiring bertambahnya waktu. Sedangkan *parameter* kecepatan angin dan tekanan udara mengalami penurunan di hari ketiga dan meningkat secara kontinu hingga 1 bulan ke depan (Supriyadi, 2019).

Berdasarkan rincian penjelasan sebelumnya, sangat dimungkinkan untuk menggunakan *Deep Learning* dengan metode *LSTM* dikarenakan mendukung kegiatan proyeksi curah hujan. Karena data pengamatan meteorologi umumnya berupa *vektor* dan *time* series, untuk itu peneliti membuat Penelitian dalam bentuk skripsi dengan judul “**PROYEKSI CURAH HUJAN DAERAH PADANG PARIAMAN MENGGUNAKAN *DEEP LEARNING* DENGAN METODE *LONG SHORT-TERM MEMORY”*.**

## Perumusan Masalah

Berdasarkan uraian pada latar belakang masalah maka yang menjadi perumusan masalah pada skripsi ini adalah sebagai berikut :

1. Bagaimana melakukan proses prediksi curah hujan dengan menggunakan pendekatan *Deep Learning* menggunakan metode *Long Short-Term Memory* dapat menghasilkan alternatif dalam pengambilan sebuah keputusan ?
2. Bagaimana penerapkan *Deep Learning* menggunakan metode *Long Short-Term Memory* dapat melakukan prediksi curah hujan.
3. Bagaimana pengujian *Deep Learning* dengan metode *Long Short-Term Memory* di implementasikan ke dalam sebuah sistem yang dibangun untuk memprediksi curah hujan di daerah padang Pariaman ?

## Hipotesis

Berdasarkan rumusan masalah yang telah ditentukan maka dapat diambil kesimpulan sebagai berikut :

1. Dengan melakukan proses prediksi curah hujan dengan menggunakan pendekatan *Deep Learning* diharapkan dapat menghasilkan alternatif dalam pengambilan sebuah keputusan.
2. Dengan menggunakan konsep pendekatan *Deep Learning* metode *Long Short-Term Memory* diharapkan dapat melakukan prediksi curah hujan.
3. Penerapan *Deep Learning* dengan metode *Long Short-Term Memory* diharapkan dapat di implementasikan ke dalam sebuah sistem yang dibangun untuk memprediksi curah hujan di daerah padang Pariaman.

## Batasan Masalah

Adapun batasan masalah pada penelitian ini adalah sebagai berikut :

1. Data yang digunakan adalah data *Stasiun Klimatologi Kelas II Sicincin Padang Pariaman* dari tahun 1985 sampai tahun 2021.
2. Metode yang digunakan dalam penelitian ini *Long Short-Term Memory*.
3. Menggunakan Bahasa Pemograman *Python*.
4. Fungsi Aktivasi yang digunakan adalah fungsi *Sigmoid (σ)* dan *Tanh*.
5. Untuk fitur variabel yang digunakan dalam penelitian ini hanya terdiri dari curah hujan (rr).
6. Data dibagi menjadi dua bagian, yaitu data training dan data test dengan rasio 9:1 di mana 9 untuk training dan 1 untuk test.

## Tujuan Penelitian

Adapun Tujuan dari penelitian ini adalah sebagai berikut :

1. Menerapkan *Deep Learning* menggunakan metode *Long Short-Term Memory* untuk melakukan proses prediksi curah hujan dalam menghasilkan alternatif dalam pengambilan sebuah keputusan.
2. Menerapkan pendekatan *Deep Learning* menggunakan metode *Long Short-Term Memory* untuk melakukan prediksi curah hujan.
3. Melakukan pengujian *Deep Learning* dengan metode *Long Short-Term Memory* di implementasikan ke dalam sebuah sistem yang dibangun untuk memprediksi curah hujan di daerah padang Pariaman.

## Manfaat Penelitian

Hasil dari penelitian di harapkan dapat memberikan manfaat sebagai berikut :

1. Menambah pengetahuan peneliti dalam memprediksi / memproyeksi data berbentuk *time series* yang di selesaikan dengan cara *Deep Learning* dengan menggunakan metode *Long Short-Term Memory*.
2. Membuktikan keakuratan metode *Long Short-Term Memory* dalam melakukan peramalan khususnya curah hujan.
3. Memberikan informasi tambahan mengenai peramalan curah hujan di daerah padang Pariaman yang akan terjadi pada masa mendatang.

## Gambaran Umum Objek Penelitian

*Stasiun Klimatologi Kelas II Sicincin Padang Pariaman* merupakan BMKG Stasiun Klimatologi Padang Pariaman yang terletak di daerah *Jalan Raya Padang – Bukittinggi KM. 51 Kapalo Hilalang Sumatera Barat*. Berikut ini adalah gambaran umum tentang Objek Penelitian :

### Sejarah BMKG

Pengamatan meteorologi dan geofisika di Indonesia dimulai pertama kali pada tahun 1841 diawali dengan pengamatan yang dilakukan secara individual oleh Dr. Pieter Loth Onnen, Kepala Rumah Sakit di Bogor. Dari tahun ke tahun kegiatannya berkembang seiring dengan meningkatnya kebutuhan akan data hasil pengamatan cuaca dan geofisika. Pada tahun 1866, pemerintah Hindia Belanda meresmikan kegiatan pengamatan p*error*angan sebagai lembaga pemerintah dengan nama Observatorium Magnetisch en Meteorologisch atau Observatorium Magnetik dan Meteorologi yang dipimpin oleh Dr. Pieter Adrian Bergsma.

Pada tahun 1879, 74 jaringan alat pengukur hujan dibangun di Jawa. pengamatan medan magnet bumi dipindahkan dari Jakarta ke Bogor Pada tahun 1902. Pada tahun 1908 pemantauan gempa dimulai dengan pemasangan komponen horizontal seismograf Wiechert di Jakarta, sedangkan pemasangan komponen vertikal dilakukan pada tahun 1928. Pada tahun 1912, pengamatan meteorologi ditata ulang dengan menambahkan jaringan sekunder. Sedangkan pada tahun 1930 jasa meteorologi mulai digunakan untuk penerangan.

Pada masa pendudukan Jepang antara tahun 1942 dan 1945, nama badan meteorologi dan geofisika diubah menjadi Kisho Kauso Kusho atau Lembaga Meteorologi. Pada tahun 1945 Setelah proklamasi kemerdekaan Indonesia, badan tersebut dibagi menjadi dua: Di Yogyakarta dibentuk Badan Meteorologi yang berkedudukan di lingkungan Mabes TNI khusus untuk melayani kepentingan Angkatan Udara. Di bawah Kementerian Pekerjaan Umum dan Energi Badan Meteorologi dan Geofisika dibentuk Di Jakarta.

Pada tanggal 21 Juli 1947 Biro Meteorologi dan Geofisika diambil alih oleh Pemerintah Belanda dan namanya diubah menjadi Meteorologisch en Geofisiche Dienst. Badan Meteorologi dan Geofisika yang dikelola oleh Pemerintah Republik Indonesia berada di Jl. Gondangdia, Jakarta. Pada tahun 1949, setelah penyerahan kedaulatan Republik Indonesia dari Belanda, Meteorologisch en Geofisiche Dienst diubah menjadi Biro Meteorologi dan Geofisika di bawah Departemen Perhubungan dan Pekerjaan Umum. Selanjutnya pada tahun 1950 Indonesia resmi masuk sebagai anggota Organisasi Meteorologi Dunia dan Kepala Badan Meteorologi dan Geofisika menjadi Wakil Tetap Indonesia dengan WMO.

Pada tahun 1955 Biro Meteorologi dan Geofisika berubah nama menjadi Lembaga Meteorologi dan Geofisika di bawah Kementerian Perhubungan, dan pada tahun 1960 namanya dikembalikan menjadi Biro Meteorologi dan Geofisika di bawah Kementerian Perhubungan Udara.

Pada tahun 1965, namanya diubah menjadi Direktorat Meteorologi dan Geofisika, posisinya tetap di bawah Kementerian Perhubungan Udara. statusnya dinaikkan menjadi lembaga setingkat eselon I dengan nama Badan Meteorologi dan Geofisika, dengan jabatan tetap di bawah Kementerian Perhubungan. Badan Meteorologi dan Geofisika.

Melalui Peraturan Presiden Nomor 61 Tahun 2008, BMG berganti nama menjadi Badan Meteorologi, Klimatologi, dan Geofisika dengan status tetap sebagai Lembaga Pemerintah Non Departemen.

### Struktur Organisasi

Berikut adalah bentuk struktur organisasi BMKG Padang Pariaman :

(Sumber : Sta. Klim. Kelas II Padang Pariaman, 2021)

Gambar 1.1. Struktur Organisasi BMKG Padang Pariaman

*Stasiun Klimatologi Kelas II Sicincin Padang Pariaman* dipimpin oleh Kepala Stasiun Klimatologi (Kaslim) bertanggung jawab terhadap seluruh bidang yang ada pada instansi BMKG, Staf yang membantu Kaslim dalam menjalankan aktivitas di kantor adalah sebagai berikut :

1. Tata usaha yang bertanggung jawab terhadap administrasi kantor.
2. Bagian analisa yang bertanggung jawab terhadap pengolahan data dan analisis data-data yang dikirim ke Balai Wilayah I.
3. Tenaga teknis yang bertanggung jawab terhadap data-data klimatologi yang ada di *Stasiun Klimatologi Kelas II Sicincin Padang Pariaman*.
4. Tim pengamat yang bertanggung jawab terhadap pengaturan jadwal pengamatan di *Stasiun Klimatologi Kelas II Sicincin Padang Pariaman*.
5. Tim komunikasi dan peralatan yang bertanggung jawab terhadap pengiriman informasi kondisi peralatan yang ada di *Stasiun Klimatologi Kelas II Sicincin Padang Pariaman*.

### Visi

Mewujudkan BMKG yang handal, tanggap dan mampu dalam rangka mendukung keselamatan masyarakat serta keberhasilan pembangunan nasional, dan berperan aktif di tingkat Internasional.

Terminologi di dalam visi tersebut dapat dijelaskan sebagai berikut :

1. Pelayanan informasi meteorologi, klimatologi, kualitas udara, dan geofisika yang handal ialah pelayanan BMKG terhadap penyajian data, informasi pelayanan jasa meteorologi, klimatologi, kualitas udara, dan geofisika yang akurat, tepat sasaran, tepat guna, cepat, lengkap, dan dapat dipertanggungjawabkan.
2. Tanggap dan mampu dimaksudkan BMKG dapat menangkap dan merumuskan kebutuhan stakeholder akan data, informasi, dan jasa meteorologi, klimatologi, kualitas udara, dan geofisika serta mampu memberikan pelayanan sesuai dengan kebutuhan pengguna jasa.

### Misi

Dalam rangka mewujudkan Visi BMKG, maka diperlukan visi yang jelas yaitu berupa langkah-langkah BMKG untuk mewujudkan Misi yang telah ditetapkan yaitu :

1. Mengamati dan memahami fenomena meteorologi, klimatologi, kualitas udara dan geofisika.
2. Menyediakan data, informasi dan jasa meteorologi, klimatologi, kualitas udara dan geofisika yang handal dan terpercaya.
3. Mengkoordinasikan dan memfasilitasi kegiatan di bidang meteorologi, klimatologi , kualitas udara dan geofisika.
4. Berpartisipasi aktif dalam kegiatan internasional di Bidang meteorologi, klimatologi, kualitas udara dan geofisika.

# BAB II LANDASAN TEORI

## Rekayasa Perangkat Lunak

IEEE Computer Society mendefinisikan Rekayasa Perangkat Lunak (RPL) sebagai penerapan suatu pendekatan yang sistematis, disiplin dan terkuantifikasi atas pengembangan, penggunaan dan pemeliharaan perangkat lunak, serta studi atas pendekatan-pendekatan ini, yaitu penerapan pendekatan engineering atas perangkat lunak (Hasanah & Untari, 2020).

RPL sendiri adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari kebutuhan pengguna, desain, penggodaan, pengujian sampai pemeliharaan sistem setelah digunakan (Hasanah & Untari, 2020).

RPL lebih fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat kepada pelanggan (customer). Adapun ilmu komputer lebih fokus pada teori dan konsep dasar perangkat komputer. Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut (Hasanah & Untari, 2020) :

1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (maintainability)
2. Dapat diandalkan dengan proses bisnis yang dijalankan perubahan yang terjadi (dependability robust)
3. Efisien dari segi sumber daya dan penggunaan
4. Kemampuan untuk dipakai sesuai dengan kebutuhan (usability)

### Proses Rekayasa Perangkat Lunak

Kerangka kerja proses membangun dasar bagi proses rekayasa perangkat lunak yang lengkap dengan cara mengidentifikasi sejumlah kecil aktivitas kerangka kerja yang cocok bagi semua proyek rekayasa perangkat lunak (Setiyani, 2018). Kerangka kerja proses pada rekayasa perangkat lunak terdiri atas lima aktivitas berikut (Setiyani, 2018) :

#### Komunikasi

Komunikasi, bertujuan untuk memahami tujuan-tujuan stakeholder atas proyek perangkat lunak yang sedang dikembangkan dan mengumpulkan kebutuhan-kebutuhan yang akan membantu mendefinisikan fitur-fitur perangkat lunak berikut dengan fungsi-fungsinya (Setiyani, 2018) :

#### Perencanaan

Kegiatan perencanaan menciptakan suatu peta yang dapat membantu membimbing tim perangkat lunak. Rencana proyek perangkat lunak menggambarkan risiko – risiko yang mungkin muncul, sumber daya yang akan dibutuhkan, produk – produk kerja yang harus dihasilkan dan schedule kerja (Setiyani, 2018) :

#### Pemodelan

Pemodelan, dilakukan bertujuan untuk membuat sketsa sehingga tim perangkat lunak dapat memahami gambaran besar produk yang akan di buat (Setiyani, 2018) :

#### Konstruksi

Konstruksi, sendiri adalah kegiatan yang menggabungkan penggodaan dan pengujian (Setiyani, 2018) :

#### Penyerahan

Penyerahan di sini merupakan Penyerahan perangkat lunak kepada user, penyajian perangkat lunak kepada user untuk di evaluasi (Setiyani, 2018) :

### Software Development Life Cycle (SDLC)

System Development Life Cycle (SDLC) adalah metodologi klasik yang digunakan untuk mengembangkan, memelihara dan menggunakan sistem informasi. Siklus hidup sistem itu sendiri merupakan metodologi, tetapi polanya lebih dipengaruhi oleh kebutuhan untuk mengembangkan sistem yang lebih cepat. Pengembangan sistem yang lebih cepat dapat dicapai dengan peningkatan siklus hidup dan penggunaan peralatan pengembangan berbasis komputer (Wahyudi, 2018).

#### Tahap-Tahap SDLC

Secara umum tahap-tahap dalam System Development Life Cycle (SDLC)terbagi dalam beberapa tahap (Wahyudi, 2018):

##### Tahap Perencanaan Sistem (system planning)

Tahap Planning Merupakan tahap awal dari pengembangan sistem, tahap ini bertujuan untuk mengidentifikasi dan memprioritaskan sistem informasi apa yang akan dikembangkan, sasaran-sasaran yang ingin dicapai, jangka waktu pelaksanaan serta mempertimbangkan dana yang tersedia dan siapa yang melaksanakan (Wahyudi, 2018).

##### Tahap Analisis Sistem (system analysis)

Tahap Analisis Sistem adalah penelitian atas sistem yang telah ada dengan tujuan untuk merancang sistem baru atau memperbaharui sistem yang sudah ada (Wahyudi, 2018).

##### Tahap Perancangan/Desain Sistem (system design)

Tahap Rancangan sistem adalah penentuan proses dan data yang diperlukan oleh sistem baru. Jika sistem ini berbasis komputer, rancangan dapat menyerta kan spesifikasi jenis peralatan yang akan digunakan (Wahyudi, 2018).

##### Tahap Penerapan/Implementasi Sistem (system implementation)

Tahap Penerapan merupakan kegiatan memperoleh dan mengintegrasikan sumber daya fisik dan konseptual yang menghasilkan suatu sistem yang bekerja. Pada tahapan ini dilakukan beberapa hal yaitu: Coding, Testing, Instalasi. Dan *Output* dari tahapan ini adalah : source code, prosedur, pelatihan (Wahyudi, 2018).

##### Tahap Pemeliharaan/Perawatan Sistem

Tahap pemeliharaan/perawatan sistem merupakan tahap yang dilakukan setelah tahap implementasi yang meliputi penggunaan sistem, audit sistem, penjagaan sistem, perbaikan sistem dan peningkatan sistem (Wahyudi, 2018).

#### Model Waterfall

Model SDLC air terjun (waterfall) sering juga disebut model sequential linier (sequential linier) atau alur hidup klasik (classic life cycle). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sequential atau terurut dimulai dari analisis kebutuhan perangkat lunak, desain, pembuatan kode program, pengujian, dan pemeliharaan (maintenance) (Tabrani & Pudjiarti, 2021). Berikut tahap tahapannya sebagai berikut (Tabrani & Pudjiarti, 2021):

##### Analisis Kebutuhan Perangkat Lunak

Proses pengumpulan kebutuhan dilakukan secara intensif untuk menspesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh user. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan (Tabrani & Pudjiarti, 2021).

##### Desain

Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka, dan prosedur penggodaan. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan (Tabrani & Pudjiarti, 2021).

##### Pembuatan Kode Program

Desain harus ditransaksikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain (Tabrani & Pudjiarti, 2021).

##### Pengujian

Pengujian fokus pada perangkat lunak secara dari segi logic dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalkan kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan (Tabrani & Pudjiarti, 2021).

##### Pendukung atau Pemeliharaan (maintenance)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke user. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru (Tabrani & Pudjiarti, 2021).

## Unified Modelling Language *(UML)*

Pemodelan dalam suatu rekayasa perangkat lunak merupakan suatu hal yang dilakukan di tahapan awal. Pemodelan dalam perangkat lunak merupakan suatu yang harus dikerjakan di bagian awal dari rekayasa, dan pemodelan ini akan mempengaruhi perkerjaan-pekerjaan dalam rekayasa perangkat lunak tersebut. Salah satu perangkat pemodelan adalah Unified Modelling Language (UML). UML merupakan salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman b*error*ientasi objek. UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan mendokumentasikan sistem perangkat lunak (Hasanah & Untari, 2020).



(Sumber : Hasanah & Untari, 2020)*.*

Gambar 2.1. Bagan UML

Berdasarkan **Gambar 2.1** berikut penjelasan singkat dari pembagian kategori tersebut (Hasanah & Untari, 2020) :

1. Behavior diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada suatu sistem.
2. Interaction diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar sub sistem pada suatu sistem.
3. Structure diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan struktur statis dari sistem yang dimodelkan.

### Behavior Diagrams

Behavior diagram adalah salah satu jenis diagram yang ada di dalam *Unified Modeling Language* (UML), di mana diagram ini digunakan untuk memberikan gambaran tingkah laku sebuah sistem informasi dan bagaimana sistem informasi tersebut melakukan tindakan terhadap kejadian atau perubahan (Sulistyo, Yudhana, & Sunardi, 2018).

#### Use Case Diagram

Use case diagram adalah teknik untuk merekam persyaratan fungsional sebuah sistem, menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Use case diagram menekankan kepada “apa” yang diperbuat oleh sistem, dan bukan “bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Seorang atau sebuah aktor adalah sebuah entitas dapat berupa manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. (Hasanah & Untari, 2020).

Tabel 2.1. Notasi Use Case Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 | *Use case* | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. |
| 2 | *Actor*  *Description: Description: actor* | Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem itu. |
| 3 | *Association* | use case yang memiliki interaksi dengan aktor. |
| 4 | *Extend* | Relasi use case tambahan ke sebuah use case di mana use case yang ditambahkan dapat berdiri sendiri. |
| 5 | *Generalization* | Relasi use case tambahan ke sebuah use case di mana use case yang ditambahkan memerlukan ini untuk menjalankan fungsinya. |

(Sumber : Hasanah & Untari, 2020)

#### Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram digunakan untuk menggambarkan langkah-langkah atau aktivitas pada suatu sistem (Hasanah & Untari, 2020).

Tabel 2.2. Notasi Activity Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 | *Initial State* | Sebuah diagram aktivitas memiliki sebuah status awal |
| 2 | *Activity* | Aktivitas yang dilakukan sistem, aktivitas *bias*anya diawali dengan kata kerja |
| 3 | *Decision* | Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu |
| 4 | *Join* | Asosiasi penggabungan di mana lebih dari satu aktivitas digabungkan menjadi satu |
| 5 | *Final State* | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir. |
| 6 | *Swimlane*    or | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi. |

(Sumber : Hasanah & Untari, 2020)

#### Statechart Diagram

Statechart diagram menelusuri individu-individu objek melalui keseluruhan daur hidupnya, menypesifikasikan semua urutan yang mungkin dari pesan-pesan yang akan diterima objek tersebut, bersama-sama dengan tanggapan atas pesan-pesan tersebut (Ilham & Fajri, 2020).

Tabel 2.3. Notasi Statechart Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 | *Initial State* | Kelas pada struktur sistem |
| 2 | *Final State* | Sama dengan konsep interface dalam beberapa pemrograman b*error*ientasi objek. |
| 3 | *Event* | Relasi antar kelas dengan makna umum.  Asosiasi *bias*anya juga disertai dengan multiplicity |
| 4 | *State* | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |

(Sumber : Ilham & Fajri, 2020).

### Interaction Diagrams

Interaction diagram merupakan diagram yang digunakan untuk menggambarkan bagaimana sebuah objek berinteraksi baik aktor dan objek sistem (Ependi, Panjaitan, & Yulianingsih, 2018).

#### *Sequence* Diagram

*Sequence* diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa message yang digambarkan terhadap waktu. *Sequence* diagram terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence* diagram *bias*a digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan *output* tertentu. Diawali dari apa yang memicu aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan (Rinaldi, 2019).

Tabel 2.4. Notasi State Machine Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 | *Actor*  *Description: Description: actor*  Or  Nama aktor | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi *bias*anya dinyatakan menggunakan kata benda di awal frasa nama actor |
| 2 | *Lifeline* | Menyatakan kehidupan suatu objek |
| 3 | *Object*  objek : kelas | Menyatakan objek yang berinteraksi pesan |
| 4 | *Active Time* | Menyatakan objek dalam keadaan aktif berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, |
| 5 | *Message type C****reate***  <<*create*>> | Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat |
| 6 | *Message type* ***Call***  1:nama\_metode() | Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri |
| 7 | *Message type* ***Send***  1: masukan | Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim |
| 8 | *Message type* ***Return***  1: keluaran  ---------------------- | Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian |
| 9 | *Message type* ***Destroy***  <<*destroy*>> | Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, jika ada create maka ada destroy. |

(Sumber : Rinaldi, 2019)

#### Collaboration Diagram

Collaboration diagram adalah cara alternatif untuk mengetahui tahap-tahap terjadinya suatu aktivitas. Perbedaan antara collaboration dan *sequence* diagram adalah collaboration diagram memperlihatkan bagaimana hubungan antara beberapa objek berdasarkan urutan dari pesan, sedangkan *sequence* diagram memperlihatkan bagaimana urutan kejadian berdasarkan waktu (Ilham & Fajri, 2020).

Tabel 2.5. Notasi Collaboration Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 | *Object*  objek : kelas | Objek yang melakukan interaksi pesan. |
| 2 | *Link* | Relasi antara objek yang menghubungkan objek satu dengan lainya atau dengan diri sendiri. |
| 4 | *State* | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |

(Sumber : Ilham & Fajri, 2020)

### Structure Diagrams

Structure diagram merupakan diagram yang digunakan untuk menggambarkan struktur dari perangkat lunak atau sistem yang dikembangkan (Ependi, Panjaitan, & Yulianingsih, 2018).

#### Class Diagram

Class diagram adalah sebuah spesifikasi yang jika di instansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain b*error*ientasi objek. Class diagram menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode/fungsi). Class diagram menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain (Rinaldi, 2019).

Tabel 2.6. Notasi Class Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 | *Class*  **nama\_kelas**  + atribut  + operasi() | Kelas pada struktur sistem |
| 2 | *Interface* | Sama dengan konsep interface dalam beberapa pemrograman b*error*ientasi objek. |
| 3 | *Association* | Relasi antar kelas dengan makna umum.  Asosiasi *bias*anya juga disertai dengan multiplicity |
| 4 | *Directed Association* | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |
| 5 | *Dependency* | Relasi antar kelas dengan ketergantungan antar kelas |
| 6 | *Aggregation* | Relasi antar kelas dengan makna semua-bagian (whole-part) |

(Sumber : Rinaldi, 2019)

#### Deployment Diagram

Menggambarkan secara lengkap bagaimana komponen di deployment dalam infrastruktur sistem, di mana komponen akan terletak, bagaimana kemampuan jaringan pada kondisi tertentu, spesifikasi server, dan hal-hal lain yang bersifat fiskal (Wijaya, Masriadi, & Ikhlas, 2020).

Tabel 2.7. Notasi Deployment Diagram

|  |  |  |
| --- | --- | --- |
| **No** | **Simbol** | **Keterangan** |
| 1 | *Package*  [Description: http://4.bp.blogspot.com/-sc_hs41py7c/VR5gbgH2yHI/AAAAAAAAAXU/xAgOjf7Nc-g/s1600/Snap%2B2015-04-03%2Bat%2B16.41.02.png](http://4.bp.blogspot.com/-sc_hs41py7c/VR5gbgH2yHI/AAAAAAAAAXU/xAgOjf7Nc-g/s1600/Snap+2015-04-03+at+16.41.02.png) | Package merupakan sebuah bungkusan dari satu atau lebih komponen. |
| 2 | *Node*  Nama\_node | *Bias*anya mengacu pada perangkat keras(hardware), perangkat lunak yang tidak dibuat sendiri (software), jika di dalam node disertakan komponen untuk mengonsistensikan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelumnya pada diagram komponen. |
| 4 | *Dependency* | Ketergantungan antar komponen, arah panah mengarah pada komponen yang dipakai. |
| 5 | *Link* | Relasi antar komponen. |

(Sumber : Wijaya, Masriadi, & Ikhlas, 2020).

## Artificial Intelligence

*Andreas Kaplan* dan *Michael Haenlein* mendefinisikan kecerdasan buatan / *Artificial Intelligence (AI)* sebagai kemampuan sistem untuk menafsirkan data eksternal dengan benar, untuk belajar dari data tersebut, dan menggunakan pembelajaran tersebut guna mencapai tujuan dan tugas tertentu melalui adaptasi yang fleksibel. Sistem seperti ini umumnya dianggap sebagai komputer (Siahaan, et al., 2020).

Lebih lanjut *Budiharto* menyatakan bahwa *Intelligence* merupakan istilah yang kompleks yang dapat didefinisikan dengan ungkapan yang berbeda seperti logika, pemahaman, self-awareness, pembelajaran, perencanaan, dan problem solving. Sedangkan “Artificial” adalah sesuatu yang tidak nyata, seperti tipuan karena merupakan hasil simulasi (Sihombing & Syaputra, 2020).

Savitri menguraikan bahwa Kecerdasan buatan / Artificial intelligence (AI) merupakan bidang ilmu komputer yang menekankan pada penciptaan mesin cerdas yang bekerja dan bereaksi seperti manusia yang perkembangannya terjadi sangat pesat di era revolusi industri keempat (Sihombing & Syaputra, 2020).

Kecerdasan diciptakan dan dimasukkan ke dalam suatu mesin / komputer agar dapat melakukan pekerjaan seperti yang dapat dilakukan oleh manusia. Beberapa macam bidang yang menggunakan kecerdasan buatan antara lain sistem pakar, permainan komputer (games), logika fuzzy, jaringan saraf tiruan dan robotika (Siahaan, et al., 2020).

Tujuan utama dari pembuatan AI adalah untuk membuat mesin memiliki fungsi yang memiliki kriteria-kriteria kecerdasan di dalamnya sehingga mesin tersebut mampu melakukan pekerjaan manusia yang lebih kompleks, tergantung dari tingkat kecerdasan AI yang digunakan (Gunova, 2021).

Dalam penerapannya, terdapat 6 kemampuan utama yang dapat diklasifikasikan sebagai AI (Gunova, 2021), Kemampuan tersebut antara lain :

* Representasi Informasi/Pengetahuan (Knowledge Representation)
* Perencanaan (Planning)
* Persepsi (Perception)
* Machine Learning / *Deep Learning*
* Pemahaman Bahasa (Natural Language Processing)
* Robotics

Meskipun 6 kemampuan di atas memiliki fungsi yang berbeda-beda, kemampuan tersebut saling berhubungan satu sama lain saat pengaplikasiannya. Bahkan hubungan antar kemampuan utama di atas akan membentuk kemampuan baru (Gunova, 2021).

## Machine Learning

Machine Learning merupakan Disiplin ilmu yang menggunakan berbagai pendekatan untuk mengajarkan komputer untuk menyelesaikan tugas-tugas di mana tidak ada algoritma yang sepenuhnya memuaskan tersedia. Dalam kasus di mana terdapat sejumlah besar jawaban potensial, satu pendekatan adalah memberi *label* beberapa jawaban yang benar sebagai valid. Ini kemudian dapat digunakan sebagai data pelatihan bagi komputer untuk meningkatkan algoritme yang digunakannya untuk menentukan jawaban yang benar. Misalnya, untuk melatih sistem untuk tugas pengenalan karakter digital, kumpulan data MNIST dari angka tulisan tangan sering digunakan (Alpaydin, 2020).

Program *Machine Learning* dapat melakukan tugas tanpa diprogram secara eksplisit untuk melakukannya. Ini melibatkan komputer belajar dari data yang disediakan sehingga mereka melakukan tugas-tugas tertentu. Untuk tugas-tugas sederhana yang ditugaskan ke komputer, dimungkinkan untuk memprogram algoritme yang memberi tahu mesin bagaimana menjalankan semua langkah yang diperlukan untuk memecahkan masalah yang dihadapi; di bagian komputer, tidak diperlukan pembelajaran. Untuk tugas yang lebih maju, mungkin sulit bagi manusia untuk membuat algoritme yang diperlukan secara manual. Dalam praktiknya, ternyata lebih efektif untuk membantu mesin mengembangkan algoritmenya sendiri, daripada meminta pemrogram manusia menentukan setiap langkah yang diperlukan (Alpaydin, 2020).

Algoritma *Machine Learning* digunakan dalam berbagai macam aplikasi, seperti dalam kedokteran, penyaringan email, pengenalan suara, dan visi komputer, di mana sulit atau tidak mungkin untuk mengembangkan algoritma konvensional untuk melakukan tugas-tugas yang diperlukan (Hu, Niu, Carrasco, Lennox, & Arvin, 2020).

Graphical user interface, text

Description automatically generated

(Sumber : Kumar, Amgoth, & Annavarapu, 2019)

Gambar 2.2. Struktur Machine learning

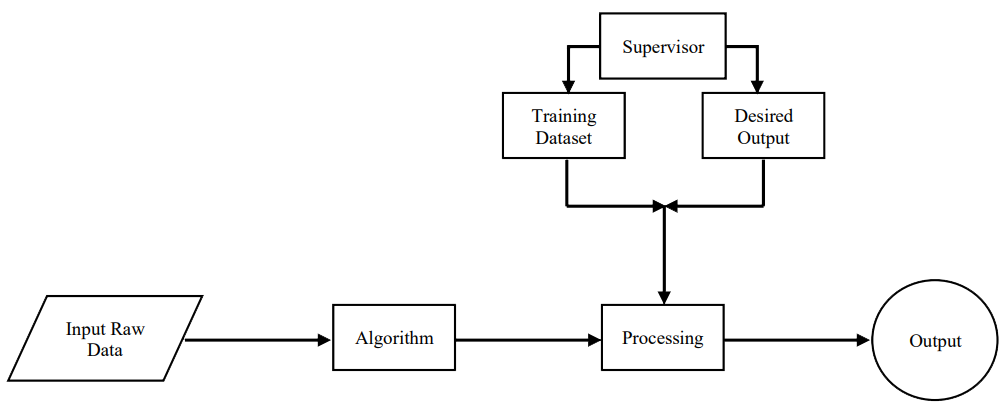
### Tipe-tipe Machine learning

Dalam Machine Learning, terdapat beberapa metode atau pendekatan yang dapat digunakan mesin untuk melakukan pembelajaran. Namun secara garis besar, terdapat 3 metode Machine Learning yang dipisahkan berdasarkan tipe *input* atau dataset dan cara pelatihannya (Gunova, 2021). Metode-metode tersebut antara lain :

#### Supervised learning

Supervised Learning (SL) merupakan sebuah metode ML di mana metode ini memberikan kumpulan data yang ber*label* dan data *input* ke dalam mesin. Maksud dari dataset yang ber*label* ini ialah untuk setiap tipe/bentuk kumpulan data yang telah diberikan, *output*nya telah ditentukan. Setelah menerima *input*, mesin kemudian akan memberikan *output* berdasarkan kumpulan data yang di*label*i. Metode ini cocok digunakan untuk penyelesaian masalah classification dan regression (Gunova, 2021).

Algoritma yang termasuk ke dalam teknik supervised learning di antaranya Decision Tree, K-Nearest Neighboor (KNN), Naive Bayes, Regresi, dan Super Vector Machine (Pamungkas, Prasetya, & Kharisudin, 2020).



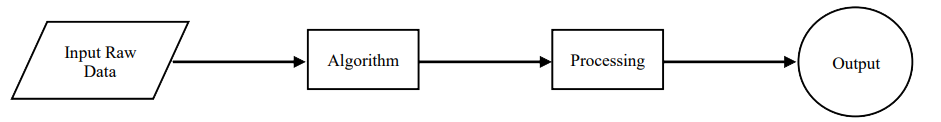
(Sumber : Gunova, 2021)

Gambar 2.3. Cara Kerja Supervised Learning (SL)

#### Unsupervised learning

Unsupervised Learning (UL) merupakan sebuah metode ML di mana metode ini hanya memberikan data *input* saja ke dalam mesin. Setelah menerima *input*, mesin kemudian akan memberikan *output* berdasarkan pola data *input* yang diterima. Metode ini cocok digunakan untuk penyelesaian masalah pengelompokan data, baik itu association maupun clustering (Gunova, 2021).

Beberapa Algoritma dalam unsupervised learning di antaranya DBSCAN, Fuzzy C-Means, K-Means, dan Self Organizing Map. DBSCAN pengelompokan berdasarkan kepadatan (density) data, konsep kepadatan menghasilkan status dari data yaitu core (inti), border (batas), dan noise (Ashari, Otniel, & Rianto, 2019).

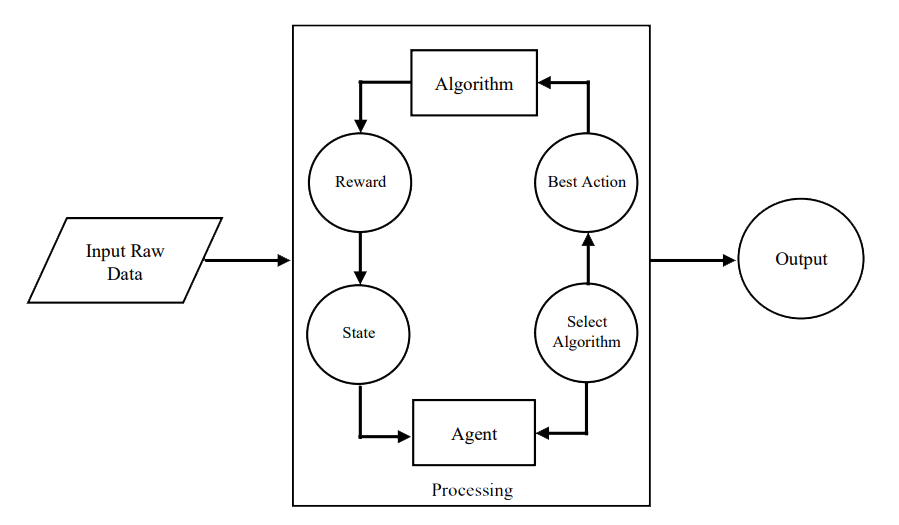


(Sumber : Gunova, 2021)

Gambar 2.4. Cara Kerja Unsupervised Learning (UL)

#### Reinforcement learning

Reinforcement machine learning adalah algoritma yang mempunyai kemampuan untuk berinteraksi dengan proses belajar yang dilakukan, algoritma ini akan memberikan poin (reward) saat model yang diberikan semakin baik atau mengurangi poin (*error*) saat model yang dihasilkan semakin buruk. Salah satu penerapan yang sering dijumpai yaitu pada mesin pencari (Fajarsari, 2020).

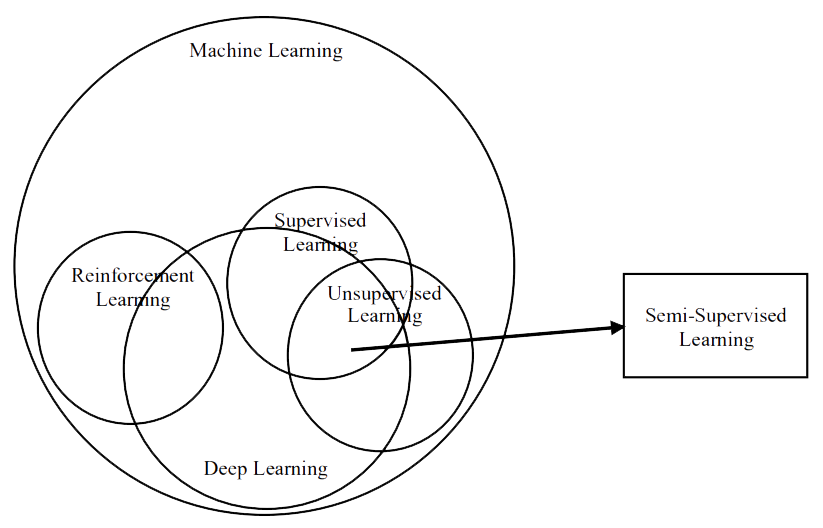


(Sumber : Gunova, 2021)

Gambar 2.5. Cara Kerja Reinforcement Learning (RL)

## *Deep Learning*

Sejak tahun 2006, *Deep Structured Learning* atau yang lebih dikenal dengan *Deep Learning* atau *Hierarchical Learning* telah muncul sebagai area baru dalam penelitian *Machine Learning* yang berdasarkan pada suatu set algoritma yang mencoba untuk memodelkan abstraksi tingkat tinggi pada data dengan menggunakan graf yang mendalam dengan beberapa *layer* pengolahan, yang terdiri dari beberapa transformasi linier dan non-linier (François, 2018).



(Sumber : Gunova, 2021)

Gambar 2.6. Salah Satu Hubungan antar metode yang dalam ML (UL, SL, RL, dan DL)

*Deep Learning* sendiri adalah cabang ilmu *machine learning* berbasis *Neural Network (NN)* atau bisa dikatakan sebagai perkembangan dari *Neural Network* (Ilahiyah & Nilogiri, 2018). Metode pendekatan *Deep Learning* mengklasifikasi data dalam dua sesi yaitu sesi training dan testing. Pada sesi training mempelajari ekstraksi fitur dari setiap data supaya bisa membedakan satu *label* dengan *label* yang lain. Pada sesi testing data-data yang diuji dapat di analisa dari hasil sesi training (Azizah, Umayah, & Fajar, 2018).



(Sumber : Savalia & Emamian, 2018)

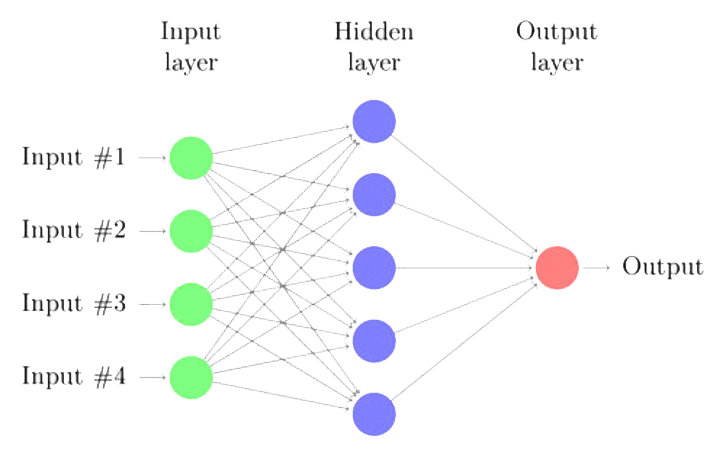
Gambar 2.7. Perbedaan Simple Neural Network dan *Deep Learning*

Dalam *Deep Learning*, sebuah komputer belajar mengklasifikasi secara langsung dari gambar atau suara. Metode *Deep Learning* menggunakan CPU dan RAM dalam proses komputasi, dan juga memanfaatkan GPU sehingga proses komputasi data yang besar dapat berlangsung lebih cepat (Ilahiyah & Nilogiri, 2018).

## Jaringan Syaraf Tiruan

*Jaringan saraf tiruan (JST) / Artificial Neural Network (ANN) / Neural Nenwork (NN)* adalah jaringan komputasi terinspirasi dari cara kerja otak manusia karena terbukti Otak manusia melakukan sesuatu hal yang sama persis seperti, hierarki pertama pada *neuron* menerima formasi pada visual cortex yang sensitif terhadap gambaran tepi dan gumpalan khusus (François, 2018).

Neural Network mempunyai lapisan masukan (*input* *layer*) dan lapisan keluaran (*output* *layer*). Pada setiap lapisan mempunyai satu atau beberapa unit *neuron*, dan mempunyai sebuah fungsi aktivasi dari unit tersebut untuk menentukan sebuah keluaran. Untuk meningkatkan kemampuan dari NN, dapat ditambahkan lapis tersembunyi atau *hidden* *layer*. Data training dapat digunakan untuk melatih NN, semakin banyak data training maka akan semakin baik unjuk kerja dari NN tersebut. Tetapi NN juga mempunyai keterbatasan pada jumlah lapisan, karena semakin banyak jumlah lapisan semakin banyaknya juga jumlah iterasi atau training yang dibutuhkan (Priyanto, Zarlis, Mawengkang, & Efendi, 2019).



(Sumber : Shekar, S´a, Ferreira, & Soares, 2018)

Gambar 2.8. Skema Jaringan Saraf Tiruan

Seperti yang terlihat pada **Gambar 2.8** Lapisan-lapisan penyusun JST tersebut dapat dibagi menjadi 3, yaitu (Satria, 2018) :

1. Lapisan *Input*, unit-unit di dalam lapisan *input* disebut unit-unit *input*. Unit-unit *input* tersebut menerima pola *input*an data dari luar yang menggambarkan suatu permasalahan (Satria, 2018)
2. Lapisan tersembunyi, unit -unit di dalam lapisan tersembunyi disebut unit- unit tersembunyi (Satria, 2018)
3. Lapisan *output*, unit-unit di dalam lapisan *output* disebut unit-unit *output* (Satria, 2018)

### Arsitektur Jaringan Syaraf Tiruan

Arsitektur atau struktur neural network adalah gambaran susunan komponen *layer* dan *neuron* pada *input*, *hidden* dan *output* yang terhubung dengan weight atau *weight*, activation function dan learning function. Perceptron dan Multi*layer* Perceptron adalah dasar dari jaringan saraf tiruan. Sebuah perceptron adalah algoritma klasifikasi biner yang dimodelkan setelah berfungsinya otak manusia, hal ini dimaksudkan untuk meniru *neuron*. Meskipun perceptron memiliki struktur sederhana tetapi memiliki kemampuan untuk belajar dan menyelesaikan masalah yang sangat kompleks. Neural Netwrok yang paling populer adalah jaringan multi perceptron feed-forward yang dilatih melalui algoritma backpropagation (Ranjit, Shrestha, Subedi, & Shakya, 2018).

A picture containing shape

Description automatically generated

(Sumber : cllau, 2020)

**Gambar 2.9. Perceptron *Input* dan *Output***

Menurut Revi, Solikhun, & Safii JST memiliki beberapa arsitektur jaringan yang sering digunakan dalam berbagai sistem. Arsitektur JST tersebut, antara lain sebagai berikut (Revi, Solikhun, & Safii, 2018) :

#### Single Perceptron

ANN yang bentuknya paling kecil disebut single perceptron yang hanya terdiri dari sebuah *neuron*, seperti terlihat pada **Gambar 2.9** (Priyanto, Zarlis, Mawengkang, & Efendi, 2019).

Single perceptron hanya terdiri dari 1 lapisan *input* dan 1 lapisan *output*. Setiap *neuron* yang terdapat di dalam lapisan *input* selalu terhubung dengan setiap *neuron* yang terdapat pada lapisan *output*. Jaringan ini hanya menerima *input* kemudian secara langsung akan mengolahnya menjadi *output* tanpa harus melalui lapisan tersembunyi (Revi, Solikhun, & Safii, 2018).

A picture containing text, night sky

Description automatically generated

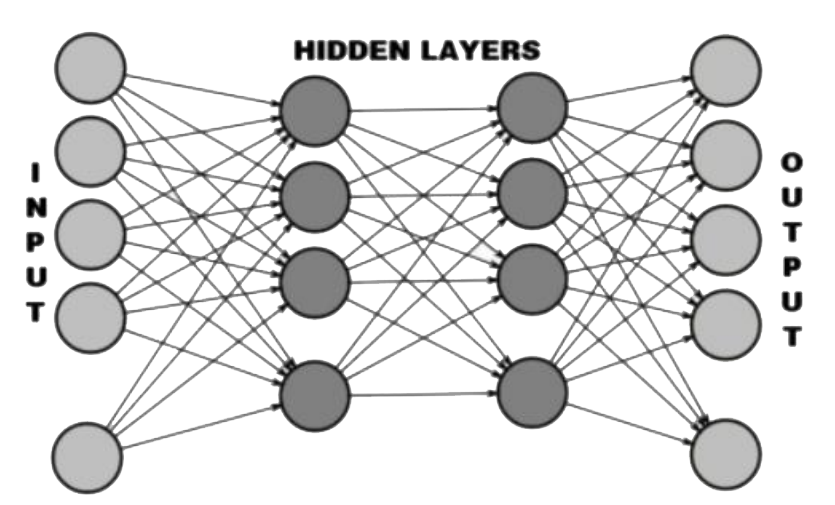
(Sumber : Putra J. G., 2020)

**Gambar 2.10. Single Perceptron**

Secara matematis pada **Gambar 2.10** terdapat *feature* vector *x* yang menjadi *input* bagi *neuron* tersebut. *Feature* vector merepresentasikan suatu data point, event atau instance. *Neuron* akan memproses *input* *x* melalui perhitungan jumlah perkalian antara nilai *input* dan *synapse weight*, yang dilewatkan pada fungsi non-linear. Pada training, yang dioptimasi adalah nilai *synapse weight* (learning *parameter*). Selain itu, terdapat juga *bias* *b* sebagai kontrol tambahan. *Output* dari *neuron* adalah hasil fungsi aktivasi dari perhitungan jumlah perkalian antara nilai *input* dan *synapse weight*. Ada beberapa macam fungsi aktivasi, misal *step function, sign function, rectifier dan sigmoid function* (Priyanto, Zarlis, Mawengkang, & Efendi, 2019).

#### Multi*layer* Perceptron (MLP)

Multi*layer* merupakan bentuk lapisan perceptron yang digabungkan, dengan menambahkan lebih banyak *layer* dan *neuron* tiap *layer*. Multi*layer* Perceptron (MLP) sendiri merupakan arsitektur yang paling banyak digunakan untuk jaringan saraf (Sen, Sugiarto, & Rochman, 2020).



(Sumber : Sen, Sugiarto, & Rochman, 2020)

Gambar 2.11. Deep Neural Network Multi*layer* Perceptron

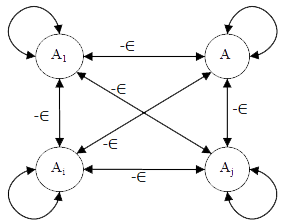
Multi*layer* Perceptron memiliki ciri khas tertentu yaitu memiliki 3 jenis lapisan yakni lapisan *input*, lapisan *output*, dan lapisan tersembunyi. Jaringan dengan banyak lapisan ini dapat menyelesaikan permasalahan yang lebih kompleks dibandingkan jaringan dengan lapisan tunggal. Namun, proses pelatihan sering membutuhkan waktu yang cenderung lama (Revi, Solikhun, & Safii, 2018).

MLP dilatih dari data pelatihan melalui proses yang disebut backpropagation. Proses ini dapat digambarkan sebagai cara untuk memperbaiki kesalahan secara progresif segera setelah terdeteksi. Pada awalnya, semua *weight* ditetapkan secara acak. Kemudian jaringan diaktifkan untuk setiap *input* dalam set pelatihan: nilai disebarkan ke depan (forward propagation) dari tahap *input* melalui tahap tersembunyi ke tahap *output* di mana prediksi dibuat (Sen, Sugiarto, & Rochman, 2020).

Karena nilai real yang diamati dalam set pelatihan diketahui, maka memungkinkan untuk menghitung kesalahan yang dibuat dalam prediksi. Proses kerja utama dalam backtracking adalah melakukan alur kembali dari *output* menuju *input* dengan menggunakan algoritma pengoptimalan yang tepat, seperti gradient descent, untuk menyesuaikan *weight* (weight) jaringan saraf dengan tujuan mengurangi kesalahan (Sen, Sugiarto, & Rochman, 2020).

#### Competitive *Layer* Net

Bentuk lapisan kompetitif merupakan jaringan saraf tiruan yang sangat besar. Interkoneksi antar *neuron* pada lapisan ini tidak ditunjukkan pada arsitektur seperti jaringan yang lain. Pada jaringan ini sekumpulan *neuron* bersaing untuk mendapatkan hak menjadi aktif atau sering pula disebut dengan prinsip *winner takes all* atau yang menanglah yang mengambil semua bagiannya (Sadli, 2018).



(Sumber : Sadli, 2018)

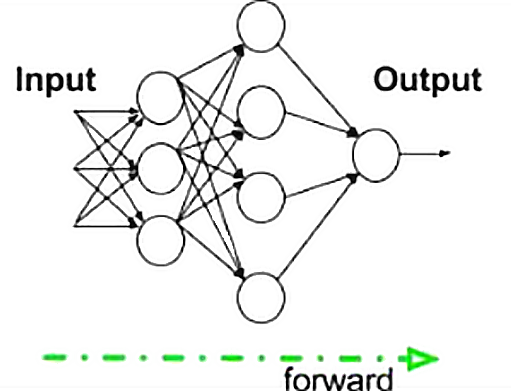
Gambar 2.12. Competitive *Layer* Net

### Teknik Training Jaringan Syaraf Tiruan

Proses Jaringan Syaraf Tiruan proses training terbagi menjadi dua yaitu Forward Pass dan Backpropagation (Winoto, 2020). Antara lain sebagai berikut :

#### Forward Propagation / Forward pass

Pemrosesan dari *layer* *input* ke *hidden* *layer* dan kemudian ke lapisan *output* adalah disebut forward propagation (Sen, Sugiarto, & Rochman, 2020). Forward propagation adalah proses mengolah sinyal *input* dengan *weight* yang tersedia pada saat melewati *Hidden* *Layer* hingga sampai ke *Output* *Layer*. Setiap *Layer* memiliki fungsi aktivasi yang berfungsi untuk mengaktifkan atau tidak suatu sinyal (Winoto, 2020).



(Sumber : Sen, Sugiarto, & Rochman, 2020)

Gambar 2.13. Forward Propagation

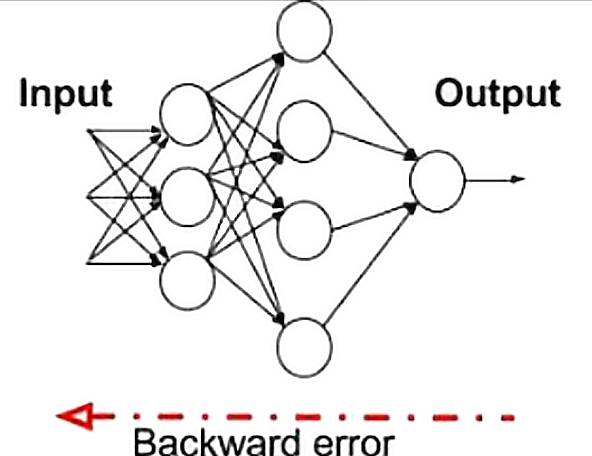
##### Algoritma Training Forward Propagation

Algoritma pembelajaran untuk JST forward propagation adalah sebagai berikut (Azise, Andono, & Pramunendar, 2019) :

1. Masing-masing *input* menerima sinyal , kemudian melanjutkan ke semua unit pada *hiden layer*
2. Pada masing-masing hiden *layer*, menjumlahkan *weight* sinyal *input* dengan persamaan : Untuk menghitung sinyal *output* pada fungsi aktivasi menggunakan persamaan
3. Masing-masing unit *output* menjumlahkan *weight* sinyal *input* dengan persamaan Untuk menghitung nilai *output*dengan menerapkan fungsi aktivasi menggunakan persamaan

#### Backward propagation / Backward Pass

Backpropagation merupakan metode pelatihan dari Artificial Neural Network yang menggunakan arsitektur multi*layer* dengan algoritma pembelajaran supervised. Metode ini bertujuan untuk melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan untuk mengenali pola dalam pelatihan dan memberikan respons yang benar terhadap pola *input* yang hampir sama dengan pola yang dipakai selama pelatihan. Metode ini juga melakukan dua tahapan, yaitu feedforward atau perhitungan maju dan backward propagation atau perhitungan mundur (Pandji, Indwiarti, & Rohmawati, 2019).



(Sumber : Sen, Sugiarto, & Rochman, 2020)

Gambar 2.14. Backward Propagation

##### Tahap Training Backward Propagation

Secara garis besar, training jaringan dengan dengan metode backpropagation meliputi tiga tahap (Satria, 2018) :

###### Tahap maju

Tahap feedforward yang dimaksud adalah proses pengolahan *input* dari pola *input* training pada *input* *layer* sampai respons yang dihasilkan mencapai *output* *layer* (Satria, 2018).

###### Tahap perhitungan error propagasi balik

Respons yang dihasilkan pada *output* *layer* akan dibandingkan dengan *output* target, kemudian dihitung *error*nya. Bila kriteria untuk kondisi berhenti (stopping condition) belum terpenuhi, maka dilanjutkan ke tahap ketiga (adjustment of the weights and *bias*es). Namun jika kondisi berhenti sudah terpenuhi, maka proses perhitungan berhenti (Satria, 2018).

###### Tahap pembaharuan weight dan bias

Kondisi ini terjadi jika *output* yang diharapkan tidak sesuai, maka jaringan akan bergerak mundur (backward) dari *output* *layer* menuju ke *input* *layer* dan akan melakukan update *weight* dan bisa serta mengulangi proses dari tahap 1. Backpropagation merupakan algoritma pembelajaran yang *bias*anya digunakan oleh perceptron dengan banyak lapisan untuk mengubah *weight*-*weight* yang terhubung dengan *neuron*-*neuron* yang ada pada lapisan tersembunyinya (Satria, 2018).

##### Algoritma Training Backward Propagation

Algoritma pembelajaran untuk JST Backpropagation adalah sebagai berikut (Muflih, Sunardi, & Yudhana, 2019) :

1. Inisiasi *weight* (tetapkan dengan nilai acak kecil)
2. Selama syarat kondisi false kerjakan langkah 2-9
3. Untuk setiap pasangan yang akan dilakukan pembelajaran, kerjakan langkah 3-8
4. Setiap unit *input* menerima sinyal dan meneruskan sinyal ke semua unit pada lapisan tersembunyi
5. Setiap unit tersembunyi menjumlahkan *weight* sinyal *input* dengan persamaan berikut: , Hitung sinyal *output* dengan fungsi aktivasinya Kirimkan sinyal ini ke semua unit pada lapisan *output*.
6. Setiap unit *output* menjumlahkan *input* ter*weight*nya Hitung sinyal *output* dengan fungsi aktivasinya
7. Setiap unit *output* menerima pola target sesuai dengan pola *input* pelatihan, kemudian hitung *error* seperti persamaan Hitung suku koreksi *weight* (digunakan untuk perbaruan wjk) Hitung suku koreksi *bias* (digunakan untuk perbaruan wok) Kirimkan ke unit-unit dilapis bawahnya
8. Setiap unit tersembunyi menjumlahkan delta *input*nya (dari unit-unit yang berada pada lapis diatasnya) Hitung suku informasi *error* Hitung suku koreksi *weight* (untuk perbaruan vij) Hitung suku koreksi *bias* (untuk perbaruan voj) Perbaruan *weight* dan *bias*
9. Setiap unit *output* perbarui *weight*-*weight* dan *bias*nya :
10. Setiap unit tersembunyi perbarui *weight*-*weight* dan *bias*nya :
11. Uji syarat berhenti, Fungsi aktivasi *sigmoid*

## *Long Short-Term Memory*

*Long Short-Term Memory (LSTM)* merupakan pengembangan dari *Recurrent Neural Network (RNN)* dengan mengatasi salah satu kekurangan *RNN* yaitu kemampuan pengelolaan informasi dalam periode yang lama yang mana dilakukan modifikasi pada Recurrent Neural Network (RNN) dengan memberi memory cell untuk dapat menyimpan informasi dalam waktu yang lama. Diusulkan oleh Sepp Hochreiter dan Jurgen Schmidhuber pada tahun 1997, *LSTM* banyak dipilih untuk prediksi berbasis waktu atau *time series* karena dikenal lebih unggul dan handal dalam melakukan prediksi dalam waktu lama dibanding algoritma lain (Zahara, Sugianto, & Ilmiddafiq, 2019; Manaswi, 2018).

A screenshot of a video game

Description automatically generated with medium confidence  
Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 2.15. Arsitektur LSTM berisi empat *layer* yang saling berinteraksi.

Kunci LSTM adalah *cell state*, garis horizontal yang melewati bagian atas diagram keadaan sel seperti ban berjalan. Ini berjalan lurus ke bawah seluruh rantai, memiliki beberapa linier kecil interaksi (Ghosh, Bose, Maji, Debnath, & Sen, 2019). Untuk setiap sel memori memiliki tiga *layer* *sigmoid* dan satu *layer* *tanh* (Qiu, Wang, & Zhou, 2019).

A picture containing text, clock

Description automatically generated

(Sumber : Manu, 2021)

Gambar 2.16. Alur Informasi *Cell state* pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Cell state*

: Nilai *Cell state* sebelum order ke *t*

Pada **Gambar 2.16**, garis horizontal yang melalui bagian atas diagram dikenal sebagai *cell state* (, ). Ini bertindak seperti ban berjalan yang berjalan di seluruh jaringan. Ini membawa informasi dari sel sebelumnya ke saat ini dan seterusnya (Hiransha, Gopalakrishnan, Menon, & Soman, 2018). Kemampuan untuk menambah atau menghapus informasi ke *cell state* dikendalikan oleh struktur yang disebut *gate*. *Gate* digunakan untuk secara opsional membiarkan informasi lewat. Informasi yang di saring melalui struktur *gate* yang akan mempertahankan dan memperbarui *cell state* memori (Ghosh, Bose, Maji, Debnath, & Sen, 2019; Qiu, Wang, & Zhou, 2019).

Graphical user interface, application

Description automatically generated

(Sumber : Manu, 2021)

Gambar 2.17. *Layer* *sigmoid* mengeluarkan angka antara nol dan satu.

Pada **Gambar 2.17** *layer* *sigmoid* mengeluarkan angka antara 0 dan 1, menggambarkan berapa banyak dari setiap komponen yang harus dilewati. Nilai 0 berarti “jangan biarkan apa pun lewat”, sedangkan nilai 1 berarti “biarkan semuanya lewat!” (Ghosh, Bose, Maji, Debnath, & Sen, 2019)*.*

### Proses Training dan Testing Pada LSTM

LSTM disebut juga sebagai jaringan saraf dengan arsitektur yang mudah beradaptasi, sehingga bentuknya dapat disesuaikan, tergantung pada aplikasinya. *Long Short-Term Memory* merupakan turunan dari metode RNN (Recurrent Neural Network). Recurrent Neural Network merupakan jaringan saraf berulang yang didesain khusus untuk menghadapi data berurutan (*sequence* data) (Wiranda & Sadikin, 2019).

#### Training Model LSTM

Beberapa tahapan dalam proses training model LSTM dengan backpropagation adalah (Arfan & ETP, 2019) :

1. Inisialisasi *weight* awal
2. *Input* data training
3. Perhitungan LSTM pada setiap *input* yaitu dimulai dengan *forget gates*, fungsi *input gates*,fungsi *cell states* dan yang terakhir fungsi *output gates*.
4. Perhitungan standard deviasi RMSE untuk mendapatkan nilai selisih antara nilai LSTM dengan target *output*.
5. Perhitungan gradien untuk menentukan nilai *weight* supaya hasil *loss* mendekati 0 dengan menggunakan Backpropagation Through Time (BTTP).
6. Setelah mendapatkan nilai gradien, maka dilanjutkan dengan persamaan fungsi optimasi dan update *weight*.
7. Kembali ke langkah dua sebanyak *epoch* yang telah ditentukan.

Bedasarkan Penjabaran di atas, Pembentukan model LSTM diawali dengan menginisialisasi paramater yang dibutuhkan yaitu *hidden* *layer* (lapisan tersembunyi), *units* (memori sel), *epoch* (putaran), dan *batch* size (jumlah sampel data). Setelah model dibentuk maka data akan dilatih dengan melewati mekanisme gates pada LSTM. Data akan dilatih terus hingga mencapai batas *error* yang diinginkan dengan penentuan serta pengubahan *parameter* yang digunakan (Agusta, Ernawati, & Muliawati, 2021).

Ketika data sudah mencapai target yang diinginkan, proses iterasi akan berhenti dan berikutnya model akan diuji dengan data pengujian atau dapat mengulang kembali proses pelatihan. Proses iterasi ini juga diolah dengan menggunakan fungsi optimasi dan *dropout*. Optimasi berguna untuk menentukan *weight* optimal dan mengurangi kesalahan sehingga dapat memaksimalkan keakuratan model. Sedangkan *dropout* berguna untuk mencegah terjadinya overfitting pada model (Agusta, Ernawati, & Muliawati, 2021).

#### Testing Model LSTM

Pengujian / Testing ini dengan mengambil data testing kemudian dibandingkan dengan data yang dihasilkan dengan metode LSTM pada rentang waktu yang ditentukan dengan metode akurasi yang digunakan menggunakan Standard deviasi RMSE (Arfan & ETP, 2019).

### Fungsi Aktivasi Pada LSTM

Fungsi aktivasi sangat berperan dalam mengaktifkan setiap *neuron* pada *jaringan saraf tiruan* serta menentukan keluaran dari suatu jaringan saraf tiruan (Susilawati & Muhathir, 2019). Berikut adalah beberapa fungsi aktivasi yang di gunakan dalam penelitian ini :

#### *Sigmoid* (σ)

Fungsi aktivasi *sigmoid* merupakan fungsi non-linear. *Input* untuk fungsi aktivasi ini berupa bilangan real dan *output* dari fungsi aktivasi ini memiliki range antara 0 sampai 1 (Suhermi, Suhartono, Dana, & Prastyo, 2018). Berikut ini perhitungan dari fungsi aktivasi *sigmoid* :

Di mana :

: data *input*

: konstanta matematika (2,718281828…)

Fungsi *sigmoid* mentransformasi range nilai dari *input* x menjadi antara 0 dan 1. Jika *inputnya* sangat negatif, maka keluaran yang didapatkan adalah 0, sedangkan jika *input* sangat positif maka nilai keluaran yang didapatkan adalah 1. Fungsi ini memiliki kekurangan yaitu dapat mematikan *gradient*, ketika aktivasi dari *neuron* mengeluarkan nilai yang berada pada range 0 atau 1, di mana gradient di wilayah ini hampir bernilai 0. Kemudian *output* dari *sigmoid* tidak zero-centered (Suhermi, Suhartono, Dana, & Prastyo, 2018).



(Sumber : Deng, Tong, Lan, & Huang, 2020)

Gambar 2.18. Illustrasi *Sigmoid*

#### Hyperbolic (*Tanh*)

Fungsi aktivasi *Tanh* merupakan fungsi non-linear. *Input* untuk fungsi aktivasi ini berupa bilangan real dan *output* dari fungsi tersebut memiliki range antara -1 sampai 1 (Suhermi, Suhartono, Dana, & Prastyo, 2018). Berikut ini perhitungan dari fungsi aktivasi *tanh* :

Di mana :

: data *input*

: konstanta matematika (2,718281828…)

Sama seperti fungsi *sigmoid*, fungsi ini memiliki kekurangan yaitu dapat mematikan *gradient*, akan tetapi fungsi ini juga memiliki kelebihan yaitu *output* yang dimiliki fungsi *Tanh* merupakan zero-centered. Dalam pengaplikasiannya fungsi *Tanh* lebih menjadi pilihan jika dibandingkan dengan fungsi *sigmoid*. Fungsi Perlu diketahui fungsi *tanh* merupakan pengembangan dari fungsi *Sigmoid* (Suhermi, Suhartono, Dana, & Prastyo, 2018).



(Sumber : Flywind, 2018)

Gambar 2.19. Ilustrasi *Tanh*

### Langkah Training Gates Pada LSTM

LSTM memiliki tiga di antaranya gerbang, untuk melindungi dan mengontrol *cell state*, Struktur gerbangnya mencakup *forget gate*, *input gate.* dan *output gate*. *Gate* terdiri dari *layer* jaring saraf *sigmoid* dan operasi perkalian pointwise (Ghosh, Bose, Maji, Debnath, & Sen, 2019; Qiu, Wang, & Zhou, 2019). Berikut ialah keterangan setiap gate yang ada pada **Gambar 2.15** :

#### Forget Gate

Pada *forget gate* informasi pada setiap data *input* yang akan diolah dan dipilih data mana saja yang akan disimpan atau dibuang pada memory cells. Fungsi aktivasi yang digunakan pada *forget gate* ini adalah fungsi aktivasi *sigmoid*. Di mana hasil keluarannya antara 0 dan 1. Jika keluarannya adalah 1 maka semua data akan disimpan dan sebaliknya jika keluarannya 0 maka semua data akan dibuang (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 2.20** :

Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 2.20. Persamaan Forget Gate pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Forget gate*

: Fungsi Aktivasi *Sigmoid*

: Nilai *Weight* untuk *Forget gate*

: Nilai *output* sebelum order ke *t*

: Nilai *input* pada order ke *t*

: Nilai *bias* pada *Forget gate*

Pada **Gambar 2.20** LSTM memutuskan informasi apa yang akan dibuang dari *cell state*. Keputusan ini dibuat oleh *layer* *sigmoid* yang disebut "*layer* forget gate." (). Terlihat pada dan dan nilai *output* antara angka 0 dan 1 untuk setiap angka dalam *cell state* pada **Gambar 2.16**. *Output* dari 1 mewakili 'sepenuhnya simpan ini' sementara 0 mewakili 'singkirkan ini sepenuhnya' (Boruah & Barman, 2018).

#### *Input* Gate

Pada *input gate* terdapat dua gates yang akan dilaksanakan, pertama akan diputuskan nilai mana yang akan diperbarui menggunakan fungsi aktivasi *sigmoid*. Selanjutnya fungsi aktivasi *tanh* akan membuat vektor nilai baru yang akan disimpan pada memory cell (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 2.21** :

Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 2.21. Persamaan yang melewati *Input* Gate pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Input gate*

: Nilai baru yang dapat ditambahkan ke *cell state*

: Fungsi Aktivasi *Sigmoid*

: Fungsi Aktivasi *Tanh*

: Nilai *Weight* untuk *Input gate*

: Nilai *Weight* untuk *Cell state*

: Nilai *output* sebelum order ke *t*

: Nilai *input* pada order ke *t*

: Nilai *bias* pada *Input gate*

: Nilai *bias* pada *cell state*

Pada langkah berikutnya di **Gambar 2.21** LSTM memutuskan informasi apa yang akan disimpan dari *cell state*. Pertama *layer* *sigmoid* yang disebut "*layer* *input gate*" () memutuskan nilai mana yang akan diperbarui. Setelah itu, *layer* membuat vektor nilai kandidat baru, , yang dapat ditambahkan ke state (Ghosh, Bose, Maji, Debnath, & Sen, 2019).

#### *Cell state* / Memory State

Pada *cell state* gates akan mengganti nilai pada memory cell sebelumnya dengan nilai memory cell yang baru. Di mana nilai ini didapatkan dari menggabungkan nilai yang terdapat pada forget gate dan *input* gate (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 2.22** :

Diagram

Description automatically generated with medium confidence

(Sumber : Manu, 2021)

Gambar 2.22. Persamaan Memperbaharui *Cell state* pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Cell state*

: *Forget gate*

: Nilai *Cell state* sebelum order ke *t*

: *Input gate*

: Kandidat konteks baru yang dapat ditambahkan ke *cell state*

Pada langkah selanjutnya di **Gambar 2.22**, kedua *layer* di gabungkan digabungkan untuk membuat pembaruan ke *cell state*. Pada Langkah inilah nilai *cell state* lama (), akan di perbaharui ke nilai dari *cell state* baru () di mana LSTM akan mengalikan *cell state* lama dengan () kemudian ditambahkan dengan () . Ini adalah nilai kandidat baru, yang diskalakan berdasarkan seberapa banyak memutuskan untuk memperbarui setiap nilai *cell state*. (Ghosh, Bose, Maji, Debnath, & Sen, 2019).

#### *Output* Gate

Pada *output gate* terdapat dua gate yang akan dilaksanakan, pertama akan diputuskan nilai pada bagian memory cell mana yang akan dikeluarkan dengan menggunakan fungsi aktivasi *sigmoid*. Selanjutnya akan ditempatkan nilai pada memory cell dengan menggunakan fungsi aktivasi *tanh*. Terakhir kedua gate tersebut di dikalikan sehingga menghasilkan nilai yang akan dikeluarkan (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 2.23** :

A picture containing text, clock

Description automatically generated

(Sumber : Manu, 2021)

Gambar 2.23. Persamaan melewati *Output* Gate pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Output gate*

: *Cell state*

: Fungsi Aktivasi *Sigmoid*

: Fungsi Aktivasi *Tanh*

: Nilai *Weight* untuk *Output gate*

: Nilai *output* sebelum order ke *t*

: Nilai *input* pada order ke *t*

: Nilai *bias* pada *Output gate*

: Nilai *output* pada order ke *t*

Terakhir pada **Gambar 2.23** adalah tahap di mana perlu memutuskan apa yang akan hasilkan. *Output* akan didasarkan pada *cell state*, tetapi akan menjadi versi yang difilter. Pertama, peneliti menjalankan *layer* *sigmoid* yang memutuskan bagian mana dari *cell state* yang akan peneliti hasilkan. Kemudian, *cell state* di tempatkan melalui *tanh* (untuk mendorong nilai menjadi antara -1 dan 1) dan mengalikannya dengan *output* *gate* *layer* *sigmoid*, sehingga hanya akan menampilkan bagian yang putuskan (Ghosh, Bose, Maji, Debnath, & Sen, 2019).

### *Loss* Function

Kinerja pembelajaran diukur dari optimalnya nilai suatu fungsi seperti minimalnya nilai *loss* dan *error*. *Loss* adalah ukuran seberapa dekat atau berbeda model yang dihasilkan dengan data asli, sedangkan *error* merupakan salah satu cara untuk menghitung *loss*. Nilai dari *loss* dan *error* tergantung dari *parameter* pembelajaran yang digunakannya. Kekurangan dari machine learning adalah membutuhkan data yang banyak untuk proses pembelajarannya (Putra J. G., 2020). Dalam Penelitian ini fungsi pengukuran nilai *error* yang akan di gunakan adalah Root Mean Squared *Error* (RMSE).

#### Root Mean Squared *Error* (RMSE)

Tingkat akurasi ditujukan selain secara visual dalam bentuk grafik juga dalam bentuk kuantitatif dengan mengukur nilai RMSE (Root Mean Square *Error*) RMSE berhubungan dengan variasi sebaran frekuensi (frequency distribution) dari besar kesalahan yang diperoleh, tapi tidak dengan variasi kesalahan. (Karno, Hastomo, Nisfiani, & Lukman, 2020). Berikut ini perhitungan dari RMSE :

Di mana (Aprian, Azhar, & Nastiti, 2020) :

: adalah jumlah data

: adalah data prediksi pada waktu ke i

: adalah data asli pada waktu i

Uji Validitas menggunakan Root Mean Squared *Error* (RMSE) dilakukan untuk mengukur hasil akurasi pengujian. RMSE merupakan salah satu contoh *parameter* yang *bias*a digunakan sebagai indikator untuk mengukur dan membandingkan kemiripan hasil prediksi dan data asli (Aprian, Azhar, & Nastiti, 2020).

### *Batch* dan *Epoch*s

*Epoch* adalah ketika seluruh kumpulan data sudah melalui proses training pada *Neural Network* sampai dikembalikan ke awal dalam satu putaran. Dalam *Neural Network* satu *epoch* itu terlalu besar dalam proses pelatihan karena seluruh data diikutkan ke dalam proses training sehingga akan membutuhkan waktu cukup lama. Agar mempermudah dan mempercepat proses training, *bias*anya data rate dibagi per *batch* (*Batch* Size).

*Batch* size merupakan jumlah sampel data yang akan disebarkan dalam sebuah *neural network*. *Batch* size efisien secara komputasi ketika berhadapan dengan kumpulan data yang besar. Penentuan nilai dari *batch* size *bias*anya tergantung peneliti dengan melihat banyak sampel (Thohari & Hertantyo, 2018).

### Normalisasi dan Denormalisasi

Dalam rangka meminimalkan *error* perlu dilakukan normalisasi. Normalisasi berfungsi untuk menghindari terjadinya berbagai anomali data dan tidak konsistensinya data. Normalisasi ini juga bertujuan untuk mengubah ukuran data menjadi lebih kecil tanpa harus mengubah data asli. Teknik normalisasi yang digunakan adalah min-max scaling. Teknik ini digunakan untuk mengatasi perbedaan nilai yang cukup besar antar kumpulan data. Cara kerjanya yakni dengan mengubah nilai pada data aktual menjadi nilai dengan skala (0,1) tanpa mengubah informasi yang ada. Teknik normalisasi dengan min-max scaling memiliki persamaan sebagai berikut (Aldi, Jondri, & Aditsania, 2018).

Di mana :

: Data hasil normalisasi

: Data asli

: Nilai minimum dari data *x*

: Nilai maximum dari data *x*

Denormalisasi adalah proses pengembalian data hasil normalisasi ke dalam data asli atau data sebenarnya. Hal tersebut dilakukan guna melihat hasil prediksi dengan cara membandingkan dengan data sebenarnya (Aldi, Jondri, & Aditsania, 2018).

Di mana :

: Hasil *output*

: Nilai dari data normalisasi

: Nilai minimal data actual keseluruhan

: Nilai maksimal data actual keseluruhan

### Interpolasi Linear

Interpolasi linier yang sering disebut sebagai interpolasi adalah kemampuan untuk menduga nilai yang terdapat di antara dua nilai lain yang dinyatakan di dalam grafik garis. Interpolasi linier merupakan salah satu metode untuk mengetahui nilai dari suatu interval dua buah titik yang terletak dalam satu garis lurus (Al Amin, Lusiana, & Hartono, 2018).

Interpolasi linier adalah cara mendapatkan nilai di antara dua data berdasarkan persamaan linier. Untuk dapat melakukannya maka minimal harus diketahui dua buah data (Al Amin, Lusiana, & Hartono, 2018).

Di mana (Al Amin, Lusiana, & Hartono, 2018). :

: Orde data yang akan di interpolation

: Orde data sebelum data yang akan di interpolasi

: Orde data sesudah data yang akan di interpolasi

: Data hasil interpolation

: Data orde sebelum data yang akan di interpolasi

: Data orde sesudah data yang akan di interpolasi

## Prediksi / Forecasting

Menurut Sucipto & Syaharuddin Prediksi / Peramalan (forecasting) adalah kegiatan mengestimasi apa yang akan terjadi pada masa yang akan datang. Peramalan diperlukan karena adanya kesenjangan waktu (timelag) antara kesadaran dibutuhkannya suatu kebijakan baru dengan waktu pelaksanaan kebijakan tersebut (Sucipto & Syaharuddin, 2018).

Menurut Putro, Furqon, & Wijoyo prediksi merupakan suatu proses untuk meramalkan atau memperkirakan suatu variabel di masa yang akan datang. Dalam kasus prediksi *bias*anya data yang sering digunakan adalah data kuantitatif. Prediksi tidak harus menghasilkan suatu jawaban yang pasti kejadian, melainkan berusaha untuk mencari jawaban yang sedekat mungkin dengan kejadian yang akan terjadi (Putro, Furqon, & Wijoyo, 2018).

Secara umum, ada dua jenis prediksi yaitu kualitatif dan kuantitatif. Prediksi kualitatif merupakan prediksi yang bersifat subjektif, hal ini karena didasarkan pada pengalaman empiris, intuisi pengambilan keputusan dan emosi manusia. Sedangkan, prediksi kuantitatif merupakan prediksi yang bersifat objektif sebab didasarkan pada data aktual dan diolah menggunakan metode tertentu (Surtiningsih, Furqon, & Adinugroho, 2018).

Jenis sesuai yang di jelaskan oleh para ahli sebelumnya prediksi yang digunakan dalam penelitian adalah prediksi kuantitatif. Dikarenakan data yang digunakan adalah data dari masa lalu berupa angka dengan runtutan waktu.

## Penelitian Terdahulu

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Pengarang, Tahun | Metode | Data | Hasil |
| 1 | (Moghar & Hamiche, 2020) | *Long Short-Term Memory* | Dokumentasi saham NYSE  (GOOGL dan NKE) dari website yahoo finance | Setelah melatih NN, hasil pengujian peneliti menunjukkan hasil yang berbeda, jumlah *epoch* serta panjang data memiliki pengaruh yang signifikan terhadap hasil pengujian. Setelah mengamati data peneliti, peneliti dapat melihat bahwa pada awalnya data kurang stabil dan memiliki nilai yang lebih rendah, setelah NKE mulai mengintip nilai yang lebih besar, aset menjadi lebih fluktuatif, maka sifat aset ini berubah. Model peneliti telah kehilangan jejak harga pembukaan sekitar 600 hingga 700 hari pengujian yang sesuai dengan perubahan sifat data. |
| 2 | (Karno, Hastomo, Nisfiani, & Lukman, 2020) | *Long Short-Term Memory* berbasis *Gated Recurrent Units* | Dokumentasi saham bank di Indonesia dari website yahoo finance | Dengan menggunakan paket library seborn di python, hasil perhitungan korelasi dapat ditunjukkan di matrik heatmap dalam bentuk numerik dan tingkatan warna. Hasil numerik korelasi berupa bilangan dengan rentang -1 dan 1, di mana nilai 1 menunjukkan hubungan yang kuat, dan nilai 0 menunjukkan hubungan yang rendah antara dua data. Jadi dapat dilihat bahwa RMSE mampu meredam perubahan kesalahan yang besar, sebaliknya MSE mampu melihat perubahan kesalahan yang kecil. GRU berupa sel yang berisi hanya 2 gate dengan rangkaian yang lebih sederhana dibandingkan dengan LSTM. |
| 3 | (Boruah & Barman, 2018) | *Long Short-Term Memory* | Identifikasi kalimat pada Novel *Bhanumati* ditulis oleh Assamese  scholar *Padmanath Gohainbaruah* | *Epoch* dengan akurasi maksimum untuk model yang berbeda diberikan. Akurasi rata-rata yang terlihat setelah setiap seribu *epoch* menampilkan kerugian rata-rata setelah setiap seribu *epoch* dari Uji 1, yang memiliki akurasi maksimum di antara model berbeda yang memiliki konfigurasi berbeda. Tetapi untuk bahasa Assam yang ditranskripsi secara fonetis pada peneliti dapat melihat bahwa akurasi rata-rata meningkat ketika peneliti meningkatkan jumlah *neuron* dari 128 menjadi 256 di setiap lapisan. Tetapi peningkatan lebih lanjut dari *neuron* menjadi 512 di setiap lapisan menurunkan akurasi. |
| 4 | (Supriyadi, 2019) | *Long Short-Term Memory* | Observasi sinoptik Stasiun  Meteorologi Maritim Tanjung Priok | Metode *Deep Learning* LSTM digunakan untuk memprediksi *parameter* cuaca, seperti suhu udara, kelembaban, kecepatan angin, dan tekanan udara. Sedangkan jumlah datanya dibagi dua menjadi training data dan test data dengan rasio 9:1.pada bulan Januari 2019. Diperoleh RMSE *parameter* suhu udara, kelembaban, kecepatan angin, dan tekanan udara nilainya semakin baik ketika menggunakan *Deep Learning* LSTM dengan update dibandingkan LSTM tanpa update. Dari *parameter* cuaca tersebut hanya *parameter* suhu dan kelembaban udara yang mengalami pertambahan RMSE seiring bertambahnya waktu. |
| 5 | (Poornima & Pushpalatha, 2019) | LSTM dengan *Weighted Linear Units* | Dokumentasi kumpulan data *China Meteorological Administration* | Dalam penelitian tersebut menyajikan Metode *Long Short-Term Memory* dengan Weighted Linear *Units* untuk memprediksi curah hujan. Neural Network dilatih dan diuji menggunakan dataset standar curah hujan. Jaringan yang dilatih akan menghasilkan atribut prediksi curah hujan. *Parameter* yang dipertimbangkan untuk evaluasi kinerja dan efisiensi model prediksi curah hujan yang diusulkan adalah Root Mean Square *Error*, akurasi, jumlah *epoch*, *loss*, dan *learning rate*. |
| 6 | (Zahara, Sugianto, & Ilmiddafiq, 2019) | *Long Short-Term Memory* | Dokumentasi secara online dari website *Dinas Perdagangan dan Perindustrian Provinsi Jawa Timur* | Pembangunan model prediksi dilakukan di lingkungan cloud computing Amazon Web Service tipe EC2. IHK prediksi pada bulan Desember 2016 yaitu 133.98, mempunyai nilai prediksi yang paling mendekati nilai IHK aktual yaitu 133.98 Dari hasil pengujian nilai RMSE tiap algoritma optimasi LSTM yang masih tergolong besar, terlihat bahwa metode LSTM belum bisa disebut metode yang maksimal dalam melakukan prediksi IHK. |
| 7 | (Aldi, Jondri, & Aditsania, 2018) | *Long Short-Term Memory* | Dokumentasi dari website *blockchain.info* | Pada penelitian tersebut dibangun model LSTM untuk memprediksi harga Bitcoin dengan pengujian *parameter* komposisi data, jumlah pola time series, jumlah *hidden* *neuron* dan max *epoch*. Pada pengujian tersebut didapatkan hasil yang terbaik yaitu dengan komposisi data latih 70% dan data uji 30%, *parameter* 1 pola time series, jumlah  25 *neuron* *hidden*, dan max *epoch* adalah 100 dengan akurasi rata-rata pada data latih 95.36% dan data testing 93.5%. |
| 8 | (Putra, Osmond, & Ansori, 2020) | *Long Short-Term Memory* | Dokumentasi data mentah dari Dinas Perindustrian dan Perdagangan Provinsi Jawa Barat | Ada 4 *parameter* yang akan diuji untuk setiap komoditas, yang terdiri dari Nilai *Epoch*, Nilai Lookback, Jumlah *Hidden* *Layer*, dan Nilai Train *Batch*. Jumlah *hidden* *layer* adalah banyaknya *layer* LSTM pada model, terdapat Single-*Layer* dan Multi-*Layer* di mana Multi-*Layer* sendiri terdiri dari 2 dan 3 *hidden* *layer*. Tujuan pengujian prediksi di sini yaitu untuk menguji tingkat akurasi dan kecocokan model LSTM dalam memprediksi harga masa depan dengan menggunakan konfigurasi dari hasil pada pengujian model. |
| 9 | (Ghosh, Bose, Maji, Debnath, & Sen, 2019) | *Long Short-Term Memory* | Dokumentasi dari website resmi *Bombay Stock Exchange* | Dalam penelitian tersebut, peneliti menganalisis pertumbuhan perusahaan dari berbagai sektor dan mencoba mencari tahu yang merupakan rentang waktu terbaik untuk memprediksi harga saham di masa depan. Prediksi bisa lebih akurat jika model akan berlatih dengan jumlah data yang lebih banyak mengatur. Kerangka kerja ini secara luas membantu dalam analisis pasar dan prediksi pertumbuhan perusahaan yang berbeda dalam rentang waktu yang berbeda. |
| 10 | (Rizki, Basuki, & Azhar, 2020) | *Long Short-Term Memory* | Dokumentasi dari website BMKG | Aplikasi berhasil memproses prediksi curah hujan kota Malang dengan *parameter* curah hujan. Jumlah *neuron* *hidden* *layer* dengan hasil paling optimal yaitu dengan 256 *neuron* *hidden* *layer*. Jumlah *epoch* dengan hasil paling optimal yaitu 150 *epoch*. Komposisi Data Train dan Data Test dengan hasil yang paling optimal yaitu dengan komposisi data train 50% dan data test 50%. Hal ini karena komposisi data train 50% dan data test 50% memiliki tingkat *error* yang paling rendah yaitu pada data train sebesar 12.079 dan pada data test sebesar 11.288. |

# DAFTAR PUSTAKA

Aditya, M. A., Mulyana, R. D., Eka, I. P., & Widianto, S. R. (2020, February). Penggabungan Teknologi Untuk Analisa Data Berbasis Data Science. (Mesran, Penyunt.) *Seminar Nasional Teknologi Komputer & Sains (SAINTEKS)*, 51-56. Diambil kembali dari https://prosiding.seminar-id.com/index.php/sainteks

Agusta, A., Ernawati, I., & Muliawati, A. (2021, August). Prediksi Pergerakan Harga Saham Pada Sektor Farmasi Menggunakan Algoritma Long Short-Term Memory. *JURNAL INFORMATIK, 17*(2).

Al Amin, I. H., Lusiana, V., & Hartono, B. (2018, November 14). PENCARIAN LINTASAN PADA COLLISION DETECTION MENGGUNAKAN PENDEKATAN INTERPOLASI LINIER. *Prosiding SINTAK, 2*, 57-61. Diambil kembali dari https://unisbank.ac.id/ojs/index.php/sintak/article/view/6513

Aldi, M. W., Jondri, & Aditsania, A. (2018, August). Analisis dan Implementasi Long Short Term Memory Neural Network untuk Prediksi Harga Bitcoin. *e-Proceeding of Engineering, 5*, 3548-3555. Diambil kembali dari https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/6739

Alpaydin, E. (2020). *Introduction to Machine Learning, fourth edition.* United States: MIT Press. Diambil kembali dari https://books.google.co.id/books?id=tZnSDwAAQBAJ

Aprian, B. A., Azhar, Y., & Nastiti, V. R. (2020, November 2). Prediksi Pendapatan Kargo Menggunakan Arsitektur Long Short Term Memory. *Jurnal Politeknik Caltex Riau, 6*, 148-157. Diambil kembali dari https://jurnal.pcr.ac.id/index.php/jkt/

Arfan, A., & ETP, L. (2019, August 22). Prediksi Harga Saham Di Indonesia Menggunakan Algoritma Long Short-Term Memory. *Seminar Nasional Teknologi Informasi dan Komunikasi, 3*(1), 225-230.

Ashari, B. S., Otniel, S. C., & Rianto. (2019). PERBANDINGAN KINERJA K-MEANS DENGAN DBSCAN UNTUK METODE CLUSTERING DATA PENJUALAN ONLINE RETAIL. *JURNAL SILIWANGI, 5*(2), 64-67.

Azise, N., Andono, P. N., & Pramunendar, R. A. (2019, July). Prediksi Pendapatan Penjualan Obat Menggunakan Metode Backpropagation Neural Networkdengan Algoritma Genetika Sebagai Seleksi Fitur. *Jurnal Teknologi Informasi-CyberKU,, 15*(2), 142-154. Diambil kembali dari http://research.pps.dinus.ac.id/index.php/Cyberku/article/view/91/82

Azizah, L. M., Umayah, S. F., & Fajar, F. (2018, November). Deteksi Kecacatan Permukaan Buah Manggis Menggunakan Metode Deep Learning dengan Konvolusi Multilayer. *SEMESTA TEKNIKA, 21*, 230-236. doi:10.18196/st.212229

Boruah, A., & Barman, P. P. (2018). A RNN based Approach for next word prediction in Assamese Phonetic Transcription. *Procedia Computer Science, 143*, 117-123. doi:10.1016/j.procs.2018.10.359

cllau, j. (2020, June 20). *Perceptron*. Diambil kembali dari Medium: https://medium.com/@cllaujhohan/perceptron-de822b401b58

Deng, X., Tong, Z., Lan, Y., & Huang, Z. (2020, May 15). Detection and Location of Dead Trees with Pine Wilt Disease Based on Deep Learning and UAV Remote Sensing. *AgriEngineering, 2*, 294-307. doi:10.3390/agriengineering2020019

Ependi, U., Panjaitan, F., & Yulianingsih, E. (2018, October 19). Pemodelan Aplikasi Mobile Sebagai Penunjang Perjalanan Wisata Menggunakan UML Diagram. *Seminar Nasional TeknologiInformasi DanKomunikasi*, 325-330.

Fajarsari, E. J. (2020). Studi Komparatif Prediksi Kestabilan Lereng Dengan Menggunakan Machine Learning (ML). *no. Ml*.

Flywind. (2018, June 23). *Graphs of the hyperbolic sine, cosine, and tangent.* Diambil kembali dari Wikipedia: https://en.wikipedia.org/wiki/File:Sinh\_cosh\_tanh\_graphs\_JCB.jpg

François, C. (2018). *Deep Learning with Python.* Manning. Diambil kembali dari https://books.google.co.id/books?id=Yo3CAQAACAAJ

Ghosh, A., Bose, S., Maji, G., Debnath, N. C., & Sen, S. (2019, September). Stock Price Prediction Using LSTM on Indian. *Proceedings of 32nd international conference, 63*, 101-110. doi:10.29007/qgcz

Gunova, V. (2021, Juni). IMPLEMENTASI DEEP LEARNING PADA SIMULASI AUTONOMOUS DRIVE MENGGUNAKAN AIRSIM. *Journal of Software Engineering, Information and Communication Technology, 2*, 83-92. Diambil kembali dari https://ejournal.upi.edu/index.php/SEICT/article/view/34674

H. M., G. E., Menon, V. K., & S. K. (2018). NSE Stock Market Prediction Using Deep-Learning Models. *Procedia Computer Science, 132*, 1351-1362. doi:10.1016/j.procs.2018.05.050

Hasanah, F. N., & Untari, R. S. (2020). *BUKU AJAR REKAYASA PERANGKAT LUNAK.* (M. Suryawinata, M. Nashrullah, & A. Y. Prajati, Penyunt.) Sidoarjo, Jawa Timur, Indonesia: UMSIDA PRESS.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., . . . Kingsbury, B. (2012, October 18). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Megazine*, 82-97. doi:10.1109/MSP.2012.2205597

Hu, J., Niu, H., Carrasco, J., Lennox, B., & Arvin, F. (2020, October 29). Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning. *IEEE Transactions on Vehicular Technology, 69*(12), 14413-14423. doi:10.1109/TVT.2020.3034800

Ilahiyah, S., & Nilogiri, A. (2018, August 2). Implementasi Deep Learning Pada Identifikasi Jenis Tumbuhan Berdasarkan Citra Daun Menggunakan Convolutional Neural Network. *Jurnal Sistem & Teknologi Informasi Indonesia, 3*, 49-56. doi:10.32528/justindo.v3i2.2254

Ilham, W., & Fajri, N. (2020, May). PENENTUAN JUMLAH PRODUKSI TAHU DENGAN MENGGUNAKAN METODE FUZZY TSUKAMOTO PADA UKM ABADI BERBASIS WEB. *JURNAL DIGIT, 10*(1), 71~82. doi:http://jurnaldigit.org/index.php/DIGIT/article/viewFile/158/116

Juanda, R. A., Jondri, & Rohmawati, A. A. (2018, August 2). Prediksi Harga Bitcoin Dengan Menggunakan Recurrent Neural Network. *e-Proceeding of Engineering, 5*, 3682-3690. Diambil kembali dari https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/6772

Karno, A. S., Hastomo, W., Nisfiani, E., & Lukman, S. (2020, August 6). Optimais Deep Learning untuk Prediksi Data Saham Di Era Pandemi Covid-19. *Seminar Nasional Teknik Elektro dan Informasi*, 43-54. doi:10.26905/santei.v1i1.3098

Knowledge@Wharton. (2018, February 13). *Vishal Sikka: Why AI Needs a Broader, More Realistic Approach*. Diambil kembali dari knowledge@wharton: https://knowledge.wharton.upenn.edu/article/ai-needs-broader-realistic-approach/

Kumar, D. P., Amgoth, T., & Annavarapu, C. S. (2019). Machine learning algorithms for wireless sensor networks: A survey. *Information Fusion, 49*, 1-25. doi:10.1016/j.inffus.2018.09.013

Manaswi, N. K. (2018). *Deep Learning with Applications Using Python.* Bangalore, Karnataka, India. doi:10.1007/978-1-4842-3516-4

Manu. (2021, January 30). *A simple overview of RNN, LSTM and Attention Mechanism*. Diambil kembali dari Start it up: https://medium.com/swlh/a-simple-overview-of-rnn-lstm-and-attention-mechanism-9e844763d07b

Moghar, A., & Hamiche, M. (2020, April). Stock Market Prediction Using LSTM Recurrent Neural Network. *International Workshop on Statistical Methods and Artificial, 170*, 1168-1173. doi:10.1016/j.procs.2020.03.049

Muflih, G. Z., Sunardi, & Yudhana, A. (2019, July). JARINGAN SARAF TIRUAN BACKPROPAGATION UNTUK PREDIKSI CURAH HUJAN DI WILAYAH KABUPATEN WONOSOBO. *MUST: Journal of Mathematics Education, Science and Technology, 4*(1), 45-46.

Pamungkas, F. S., Prasetya, B. D., & Kharisudin, I. (2020). Perbandingan Metode Klasifikasi Supervised Learning pada Data Bank Customers Menggunakan Python. *PRISMA, Prosiding Seminar Nasional Matematika*, 689-694. Diambil kembali dari https://journal.unnes.ac.id/sju/index.php/prisma/

Pandji, B. Y., Indwiarti, & Rohmawati, A. A. (2019, September). PERBANDINGAN PREDIKSI HARGA SAHAM DENGAN MODEL ARIMA DAN ARTIFICIAL NEURAL NETWORK. *Indonesia Journal on Computing (Indo-JC), 4*(2), 189-198. doi:10.34818/INDOJC.2019.4.2.344

Pardede, J., & Ibrahim, R. G. (2020, December 31). Implementasi Long Short-Term Memory Untuk Identifikasi Berita Hoax Berbahasa Inggris Pada Media Sosial. *J-COSINE, 4*, 179-187. doi:10.29303/jcosine.v4i2.361

Poornima, S., & Pushpalatha, M. (2019, October 31). Prediction of Rainfall Using Intensified LSTM Based Recurrent Neural Network with Weighted Linear Units. *Atmosphere, 10*. doi:10.3390/atmos10110668

Priyanto, D., Zarlis, M., Mawengkang, H., & Efendi, S. (2019, December). Studi Literatur Konsep Dasar Machine Learning Dan Neural Network. *Seminar Nasional Matematika dan Terapan 2019, 1*, 160-166.

Puspaningrum, A., Bunga, M. S., & Iryanto. (2020, November). Klasifikasi Perubahan Perangkat Lunak pada Mobile App Review dengan Menggunakan Metode Long Short Term Memory (LSTM). *Jurnal IKRA-ITH Informatika, 3*, 41-46.

Putra, J. G. (2020). *Pengenalan Konsep Pembelajaran Mesin dan Deep Learning.* Tokyo, Jepang. Diambil kembali dari https://wiragotama.github.io/resources/ebook/intro-to-ml-secured.pdf

Putra, M. R., Osmond, A. B., & Ansori, A. S. (2020, April 1). ESTIMASI HARGA KEBUTUHAN POKOK DI KOTA BANDUNG DAN PROVINSI JAWA BARAT MENGGUNAKAN METODE LSTM. *e-Proceeding of Engineering, 7*, 1455-1459.

Putro, B., Furqon, M. T., & Wijoyo, S. H. (2018, November). Prediksi Jumlah Kebutuhan Pemakaian Air Menggunakan Metode Exponential Smoothing (Studi Kasus : PDAM Kota Malang). *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2*, 4679-4686. Diambil kembali dari http://j-ptiik.ub.ac.id

Qiu, J., Wang, B., & Zhou, C. (2019, December 13). Forecasting stock prices with long-short term memory neural network based on attention mechanism. (T. Song, Penyunt.) *PLoS ONE*, 1-15. doi:10.1371/journal.pone.0227222

Ranjit, S., Shrestha, S., Subedi, S., & Shakya, S. (2018, July 01). Foreign Rate Exchange Prediction Using Neural Network and Sentiment Analysis. *Proceedings - IEEE 2018 International Conference on Advances in Computing, Communication Control and Networking, ICACCCN 2018*, 1173–1177. doi:10.1109/ICACCCN.2018.8748819

Revi, A., Solikhun, & Safii, M. (2018, October). JARINGAN SYARAF TIRUAN DALAM MEMPREDIKSI JUMLAH PRODUKSI DAGING SAPI BERDASARKAN PROVINSI. *KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer), 2*(1), 297-304. Diambil kembali dari http://ejurnal.stmik-budidarma.ac.id/index.php/

Rinaldi, R. (2019, February 26). PENERAPAN UNIFIED MODELLING LANGUAGE (UML) DALAM ANALISIS DAN PERANCANGAN APLIKASI E-LEARNING. *Jurnal SIMTIKA, 2*, 43–50. Diambil kembali dari http://ejournal.undhari.ac.id/index.php/simtika/article/view/15

Rizki, M., Basuki, S., & Azhar, Y. (2020, March). Implementasi Deep Learning Menggunakan Arsitektur Long Short Term Memory Untuk Prediksi Curah Hujan Kota Malang. *REPOSITOR, 2*(3), 331-338.

Sadli, A. (2018, April). SIMULASI PENGENALAN KARAKTER MENGGUNAKAN NEURAL NETWORK PADA MATLAB. *JURNAL SISTEM INFORMASI DAN TEKNOLOGI INFORMASI, 7*(1), 89-97.

Satria, B. (2018, December 15). Prediksi Volume Penggunaan Air PDAM Menggunakan Metode Jaringan Syaraf Tiruan Backpropagation. *JURNAL RESTI, 2*(3), 674-684.

Savalia, S., & Emamian, V. (2018, April 28). Cardiac Arrhythmia Classification by Multi-Layer Perceptron and Convolution Neural Networks. *Bioengineering, 5*, 2-10. doi:10.3390/bioengineering5020035

Schneiderman, H., & Kanade, T. (2002, August 06). A statistical method for 3D object detection applied to faces and cars. *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*. doi:10.1109/CVPR.2000.855895

Sen, S., Sugiarto, D., & Rochman, A. (2020). Komparasi Metode Multilayer Perceptron (MLP) dan Long Short Term Memory (LSTM) dalam Peramalan Harga Beras. *target, 4*(1).

Setiyani, L. (2018). *Rekayasa Perangkat Lunak [Software Engineering].* (L. Setiyani, Penyunt.) Karawang, Jawa Barat, Indonesia: Jatayu Catra Internusa.

Shekar, A. K., S´a, C. R., Ferreira, H., & Soares, C. (2018, April). Building robust prediction models for defective sensor data using Artificial Neural Networks.

Siahaan, M., Jasa, C. H., Anderson, K., Rosiana, M. V., Lim, S., & Yudianto, W. (2020, November). Penerapan Artificial Intelligence (AI)Terhadap Seorang Penyandang Disabilitas Tunanetra. *Journal of Information System and Technology, 1*, 186-193. Diambil kembali dari https://journal.uib.ac.id/index.php/joint/article/view/4322/1122

Sihombing, E. N., & Syaputra, M. Y. (2020, September 23). IMPLEMENTASI PENGGUNAAN KECERDASAN BUATAN DALAM PEMBENTUKAN PERATURAN DAERAH. *JURNAL ILMIAH KEBIJAKAN HUKUM, 4*, 419-434. doi:10.30641/kebijakan.2020.V14.419-434

Sta. Klim. Kelas II Padang Pariaman. (2021, November). Padang Pariaman, Sumatera Barat, Indonesia: BMKG Klimatologi Padang Pariaman.

Sucipto, L., & Syaharuddin, S. (2018, September 2). Konstruksi Forecasting System Multi-Model untuk pemodelan matematika pada peramalan Indeks Pembangunan Manusia Provinsi Nusa Tenggara Barat. *Jurnal Ilmiah Teknologi Sistem Informasi, 4*, 114-123. doi:10.26594/register.v4i2.1263

Suhermi, N., Suhartono, Dana, I. G., & Prastyo, D. D. (2018, November). Pemilihan Arsitektur Terbaik pada Model Deep Learning Melalui Pendekatan Desain Eksperimen untuk Peramalan Deret Waktu Nonlinier. *Departemen Statistika, 18*, 153 –159. doi:10.29313/jstat.v18i2.4545

Sulistyo, A., Yudhana, A., & Sunardi, S. (2018). PERANCANGAN SISTEM INFORMASI BREEDING PLACE TERHADAP KEJADIAN DEMAM BERDARAH DENGUE (DBD) BERBASIS MOBILE TECHNOLOGY. *Prosiding SNST Fakultas Teknik, 1*(1).

Supriyadi, E. (2019, Oktober 16). PREDIKSI PARAMETER CUACA MENGGUNAKAN DEEP LEARNING LONG-SHORT TERM MEMORY (LSTM). *JURNAL METEOROLOGI DAN GEOFISIKA, 21*, 55-67. Diambil kembali dari https://pdfs.semanticscholar.org/f17a/634eb087b317c0e9eaf632c0ec4350725eb3.pdf

Surtiningsih, L., Furqon, M. T., & Adinugroho, S. (2018, August 8). Prediksi Jumlah Kunjungan Wisatawan Mancanegara Ke Bali Menggunakan Support Vector Regressiondengan Algoritma Genetika. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer, 2*, 2578-2586. Diambil kembali dari https://j-ptiik.ub.ac.id/index.php/j-ptiik/article/view/1857

Susilawati, & Muhathir. (2019, January 2). Analisis Pengaruh Fungsi Aktivasi, Learning Rate Dan Momentum Dalam Menentukan Mean Square Error (MSE) Pada Jaringan Saraf Restricted Boltzmann Machines (RBM). *JITE (Journal of Informatics and Telecommunication Engineering), 2*, 77-91.

Swarnkar, A., & Swarnkar, A. (2019). *Artificial Intelligence Based Optimization Techniques: A Review.* Singapore, Singapore, Singapore: Springer. doi:https://doi.org/10.1007/978-981-15-0214-9\_12

Tabrani, M., & Pudjiarti, E. (2021, August 2). Pembangunan Perangkat Lunak E-Learning dalam Kegiatan Belajar Mengajar Sekolah Menengah Kejuruan Angkasa Husein. *Jurnal Sains Manajemen Informatika dan Komputer, 20*(2), 130~138. Diambil kembali dari https://ojs.trigunadharma.ac.id/

Tarkus, E. D., Sompie, S. R., & Jacobus, A. (2020, Juni 30). Implementasi Metode Recurrent Neural Network pada Pengklasifikasian Kualitas Telur Puyuh. *Jurnal Teknik Informatika, 15*, 137-144.

Thohari, A. N., & Hertantyo, G. B. (2018, August 11). Implementasi Convolutional Neural Network untuk Klasifikasi Pembalap MotoGP Berbasis GPU. *Conference on Electrical Engineering, Telematics, Industrial Technology, and Creative Media (CENTIVE)*, 50-55. Diambil kembali dari http://conferences.ittelkom-pwt.ac.id/index.php/centive/article/view/9

Wahyudi, A. (2018). Perancangan sistem menggunakan metode sdlc. *J. Din. Inform, 4*(2), 1-7.

Wibisono, Y., & Khodra, M. L. (2018, April 09). Pengenalan Entitas Bernama Otomatis untuk Bahasa Indonesia dengan Pendekatan Pembelajaran Mesin. doi:http://dx.doi.org/10.31227/osf.io/vud2p

Wijaya, R., Masriadi, & Ikhlas, M. (2020, July). IMPLEMENTASI MODEL WATERFALL PADA PENGEMBANGAN SISTEM INFORMASI PERHITUNGAN NILAI MATA PELAJARAN BERBASIS WEB PADA SEKOLAH DASAR AL-AZHAR SYIFA BUDI JATIBENING. *INFORMATION SYSTEM DEVELOPMENT, 5*(2).

Winoto, A. R. (2020, September). RANCANG BANGUN APLIKASI PRESENSI DENGAN MEDIA SUARA MENGGUNAKAN MFCC DAN ANN BERBASIS ANDROID. *Jurnal Ilmiah Sainsbertek Jurnal Ilmiah Sains & Teknologi, 1*, 46--57.

Wiranda, L., & Sadikin, M. (2019, December). PENERAPAN LONG SHORT TERM MEMORY PADA DATA TIME SERIES UNTUK MEMPREDIKSI PENJUALAN PRODUK PT. METISKA FARMA. *Jurnal Nasional Pendidikan Teknik Informatika, 8*(3), 184-196.

Zahara, S., Sugianto, & Ilmiddafiq, M. B. (2019, December 2). Prediksi Indeks Harga Konsumen Menggunakan Metode Long Short Term Memory(LSTM) Berbasis Cloud Computing. *JURNAL RESTI (Rekayasa Sistem dan Teknologi Informasi), 3*, 357–363. doi:10.29207/resti.v3i3.1086

Zhou, V. (2019, March 6). *Machine Learning for Beginners: An Introduction to Neural Networks*. Diambil kembali dari towards data science: https://towardsdatascience.com/machine-learning-for-beginners-an-introduction-to-neural-networks-d49f22d238f9