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

**SKRIPSI**

*Untuk Memenuhi Salah Satu Syarat*

*Memperoleh Gelar Sarajana Komputer*

**Program Studi : Tehnik 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 II LANDASAN TEORI

## 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 variable di masa yang akan datang. Dalam kasus prediksi biasanya 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.

## Time Series

Time series adalah rangkaian data yang berupa nilai pengamatan yang diukur selama kurun waktu tertentu, berdasarkan waktu dengan interval yang sama, dalam hal ini adalah data nilai indeks harga saham yang diperoleh dalam jangka waktu yang berurutan data ini dikumpulkan dari waktu ke waktu. Waktu yang digunakan dapat berupa hari, bulan, tahun (Pandji, Indwiarti, & Rohmawati, 2019).

Analysis *time series* dilakukan untuk memperoleh pola dari data tersebut dengan menggunakan historical data yang akan digunakan untuk memprediksi suatu nilai di masa yang akan datang. Oleh karena itu untuk memilih suatu metode time series yang tepat harus mempertimbangkan pola data, agar metode yang digunakan dengan pola tersebut dapat diuji (Pandji, Indwiarti, & Rohmawati, 2019).

## Perangkat Lunak *(Software)*

Software / Perangkat Lunak komputer adalah sekumpulan data elektronik yang disimpan dan diatur oleh komputer. Data elektronik yang disimpan oleh komputer itu dapat berupa program atau instruksi yang akan menjalankan suatu perintah. Melalui software atau perangkat lunak inilah suatu komputer dapat menjalankan suatu perintah (Wali, 2020).

### Jenis-Jenis Perankat Lunak

Software atau perangkat lunak komputer berdasarkan distribusinya dibedakan menjadi beberapa macam, yaitu software berbayar, software gratis atau free (freeware, free software, shareware, dan adware) (Wali, 2020).

#### Software Berbayar

Software berbayar merupakan perangkat lunak yang didistribusikan untuk tujuan komersial. Setiap pengguna yang ingin menggunakan atau mendapatkan software tersebut dengan cara membeli atau membayar pada pihak yang mendistribusikannya. Pengguna yang menggunakan software berbayar umumnya tidak diijinkan untuk menyebarluaskan software tersebut secara bebas tanpa izin dari penerbitnya. Contoh software berbayar ini misalnya adalah sistem Microsoft Windows, Microsoft Office, Adobe Photoshop, dan lain-lain (Wali, 2020).

#### Freeware

Freeware atau perangkat lunak gratis adalah perangkat lunak komputer berhak cipta yang gratis digunakan tanpa batasan waktu, berbeda dari shareware yang mewajibkan penggunanya membayar (misalnya setelah jangka waktu percobaan tertentu atau untuk memperoleh fungsi tambahan). Freeware juga didefinisikan sebagai program apa pun yang didistribusikan gratis, tanpa biaya tambahan. Sebuah contoh utama adalah Suite Browser, Mail Client, dan Mozilla News, juga didistribusikan di bawah GPL (Wali, 2020).

#### Free Software

Free software lebih mengarah kepada bebas penggunaan, tetapi tidak harus gratis. Pada kenyataannya, namanya adalah karena bebas untuk mencoba perangkat lunak sumber terbuka (open source) dan di sanalah letak inti dari kebebasan (Wali, 2020).

Perbedaan yang nyata antara free software dan freeware. Konflik muncul dalam arti kata free dalam Bahasa Inggris, yang berarti keduanya bebas dan gratis. Oleh karena itu, seperti yang disebutkan sebelumnya, free software tidak perlu bebas, sama seperti freeware tidak harus gratis (Wali, 2020).

#### Shareware

Shareware juga bebas, tetapi lebih dibatasi untuk waktu tertentu. Shareware adalah program terbatas didistribusikan baik sebagai demonstrasi atau versi evaluasi dengan fitur atau fungsi yang terbatas atau dengan menggunakan batas waktu yang ditetapkan (misalnya, 30 hari) (Wali, 2020).

#### Malware

Perangkat perusak (Bahasa Inggris: malware, berasal dari lakuran kata malicious dan software) adalah perangkat lunak yang diciptakan untuk menyusup atau merusak sistem komputer, peladen, atau jejaring komputer tanpa izin termaklum (informed consent) dari pemilik. Istilah ini adalah istilah umum yang dipakai oleh pakar komputer untuk mengartikan berbagai macam perangkat lunak atau kode perangkat lunak yang mengganggu atau mengusik. Istilah “virus computer” kadang-kadang dipakai sebagai frasa pemikat (catch phrase) untuk mencakup semua jenis perangkat perusak, termasuk virus murni (true virus) (Wali, 2020).

### 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, disain, pengkodean, 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 computer lebih fokus pada teori dan konsep dasar perangkat computer. 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.

#### Perencanaan

Kegiatan perencanaan menciptkan 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.

#### Pemodelan

Pemodelan, dilakukan bertujuan untuk membuat sketsa sehingga tim perangkat lunak dapat memahami gambaran besar produk yang akan di buat.

#### Konstruksi

Konstruksi, sediri adalah kegiatan yang menggabungkan pengkodean dan pengujian.

#### Penyerahan

Penyerahan di sini merupakan Penyerahan perangkat lunak kepada user, penyajian perangkat lunak kepada user untuk di evaluasi.

## Unified Modelling Language *(UML)*

Pemodelan dalam suatu rekayasa perangkat lunak merupakan suatu hal yang dilakukan di tahapan awal. Pemodelan delam 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 standart bahasa yang banyak digunakan di dunia industry untuk mendefinisikan requirement, membuat analisis & desain, sert amenggambarkan arsitektur dalam pemrograman berorientasi 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. Bagan UML

Berdasarkan **Gambar 2** 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 subsistem pada suatu sistem.
3. Structure diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan strutur statis dari sistem yang dimodelkan.

### Behavior Diagrams

#### Use Case Diagram

Use case diagram adalah teknik untuk merekam persyaratan fungsional sebuah system, 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 actor dengan sistem. Seorang atau sebuah aktor adalah sebuah entitas dapat berupa manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu. (Hasanah & Untari, 2020).

Table 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 dimana use case yang ditambahkan dapat berdiri sendiri. |
| 5 | *Generalization* | Relasi use case tambahan ke sebuah use case dimana 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 langkahlangkah atau aktivitas pada suatu sistem (Hasanah & Untari, 2020).

Table 2. Notasi Activity Diagram

|  |  |  |
| --- | --- | --- |
| No | Simbol | Keterangan |
| 1 | *Initial State* | Sebuah diagram aktivitas memiliki sebuah status awal |
| 2 | *Activity* | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja |
| 3 | *Decision* | Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu |
| 4 | *Join* | Asosiasi penggabungan dimana 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)

#### State Machine Diagram

State Machine Diagram menggambarkan perilaku objek tertentu dalam sistem perangkat lunak, menentukan urutan peristiwa dan tanggapan terhadap peristiwa objek selama masa berlakunya. Diagram mesin keadaan umum terdiri dari himpunan keadaan berhingga, himpunan keadaan semu berhingga, dan himpunan transisi berhingga (Damjan & Vatanawood, 2017).

Table 3. Notasi State Machine Diagram

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

(Sumber : Damjan & Vatanawood, 2017)

### Interaction Diagrams

#### Sequence Diagram

Sequence diagrammenggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa messageyang digambarkan terhadap waktu. Sequence diagramterdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Sequence diagrambiasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah eventuntuk menghasilkan outputtertentu. Diawali dari apa yang men-triggeraktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan outputapa yang dihasilkan (Rinaldi, 2019).

Table 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 biasanya 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 atau diagram kolaborasi merupakan suatu diagram yang memperlihatkan juga mengelompokkan pesan yang terdapat pada sekitar objek (Supriatna & Sherin, 2017).

Table 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 antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |

(Sumber : Supriatna & Sherin, 2017)

### Structure Diagrams

#### Class Diagram

Class diagram adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class diagrammenggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). Class diagrammenggambarkan struktur dan deskripsi class, packagedan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain (Rinaldi, 2019).

Table 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 berorientasi objek. |
| 3 | *Association* | Relasi antarkelas dengan makna umum.  Asosiasi biasanya juga disertai dengan multiplicity |
| 4 | *Directed Association* | Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain. |
| 5 | *Dependency* | Relasi antar kelas dengan kebergantungan antar kelas |
| 6 | *Aggregation* | Relasi antarkelas dengan makna semua-bagian (whole-part) |

(Sumber : Rinaldi, 2019)

#### Deployment Diagram

Bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (run-time). Memuat simpul-simpul beserta komponen-komponen yang ada di dalamnya. Diagram deployment berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponenkomponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (distributed computing) (Dermawan & Hartini, 2017).

Table 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 | Biasanya mengacu pada perangkat keras(hardware), perangkat lunak yang tidak dibuat sendiri (software), jika didalam node disertakan komponen untuk mengkonsistensikan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah didefinisikan sebelumnya pada diagram komponen. |
| 4 | *Dependency* | Kebergantungan antar komponen, arah panah mengarah pada komponen yang dipakai. |
| 5 | *Link* | Relasi antar komponen. |

(Sumber : Dermawan & Hartini, 2017).

## Bahasa Pemograman Python

Python adalah bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif. Python juga didukung oleh komunitas yang besar (Syahrudin & Kurniawan, 2018).

Python mendukung multi paradigma pemrograman, utamanya, namun tidak dibatasi pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang tersedia pada python adalah sebagai bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai bahasa script meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa script. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi (Syahrudin & Kurniawan, 2018). Saat ini kode python dapat dijalankan di berbagai platform sistem operasi, beberapa diantaranya adalah (Syahrudin & Kurniawan, 2018) :

* Linux/Unix
* Windows
* Mac OS X
* Java Virtual Machine
* Amiga
* Palm
* Symbian (untuk produk-produk Nokia)

Python didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi. Namun pada prinsipnya Python dapat diperoleh dan dipergunakan secara bebas, bahkan untuk kepentingan komersial. Lisensi Python tidak bertentangan baik menurut definisi Open Source maupun **General Public License (GPL)** (Syahrudin & Kurniawan, 2018)**.**

## Database

Database adalah sebuah system yang di buat untuk mengorganisasi, menyimpan dan menarik data dengan mudah. Database terdiri dari kumplan data yang terorganisir untuk 1 atau lebih penggunaan, dalam bentuk digital. Database digital di manage menggunakan Database Management System (DBMS), yang menyimpan isi database, mengizinkan pembuatan dan maintenance data dan pencarian dan akses yang lain Beberapa Database yang ada saat ini adalah, Mysql, Sql Server, Ms.Access, Oracle, dan PostgreSql (Ramadhan & Mukhaiyar, 2020).

### Fungsi Database

Adapun beberapa fungsi dari database adalah (Ramadhan & Mukhaiyar, 2020) :

1. Mempermudah identifikasi data dengan cara pengelompokkan data, salah satu contoh nya dengan pembuatan beberapa tabel atau field yang berbeda-beda.
2. Meminimalisir suatu data ganda.
3. Mempermudah penggunaan user dalam berbagai hal, misalnya pada saat peng-inputan data baru.
4. Penyimpanan secara digital
5. Menjadi alternatif lain terkait masalah penyimpanan ruang dalam suatu aplikasi.

### Jenis Database

Adapun beberapa jenis database, diantaranya (Ramadhan & Mukhaiyar, 2020) :

1. Operational Database, Database jenis ini mengoperasikan penyimpanan data yang sangat rinci agar dapat dengan mudah digunakan. Database ini biasa digunakan untuk database pelanggan
2. Relational Database Pada database ini, user dapat mengakses atau mencari informasi dalam tabel yang berbeda –beda.
3. Distributed DatabaseUntuk database jenis ini dapat mendistribusikan data –data secara tersebar namun saling berhubungan serta dapat diakses secara bersama –sama.
4. External Database Database terakhir ini sering digunakan sebagai keperluan komersial karena kemudahan mengaksesnya yang memang dikhususkan untuk publik.

### MySQL

SQL (Structured Query Language) adalah sebuah bahasa scripting yang dipergunakan untuk mengolah database. Database besar seperti MySQL, PostgreSQL dan SQL Server sudah menggunakan SQL untuk mengolah database nya. SQL yangdi pergunakan software database tersebut adalah sama kecuali sedikit perbedaan dibeberapa tempat (Ramadhan & Mukhaiyar, 2020).

MySQL menggunakan bahasa SQL untuk mengakses database nya. Lisensi Mysql adalah FOSS License Exception dan ada juga yang versi komersial nya. Tag Mysql adalah “The World's most popular open source database”. MySQL tersedia untuk beberapa platform, di antara nya adalah untuk versi windows dan versi linux (Ramadhan & Mukhaiyar, 2020).

## Normalisasi dan Denormalisasi

### Normalisasi

Dalam rangka meminimalkan error perlu dilakukan normalisasi. Normalisasi berfungsi untuk menghindari terjadinya berbagai anomali data dan tidak konsistensinya data. Normalisasi ini juga bertujuan untuk merubah ukuran data menjadi lebih kecil tanpa harus merubah data asli. Teknik normalisasi yang digunakan adalah min-max scaling. Teknik ini digunakan untuk mengatasi perbedaan nilai yang cukup besar antar dataset. 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).

Dimana :

: Data hasil normalisasi

: Data asli

: Nilai minimum dari data *x*

: Nilai maximum dari data *x*

### Denormalisasi

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).

Dimana :

: Nilai dari data noermalisasi

: Hasil output

: Nilai minimal data actual keseluruhan

: Nilai maksimal data actual keseluruhan

## Fungsi Aktivasi

Fungsi aktivasi sangat berperan dalam mengaktifkan setiap neuron pada *jaringan saraf tiruan* serta menentukan keluaran dari suatu jaringan saraf tiruan (Susilawati & Muhathir, 2019). Fungsi aktivasi berfungsi untuk menghitung nilai keluaran berdasarkan nilai *input* dan bobot pada neuron, fungsi optimisasi berfungsi untuk meminimalisir error antara nilai keluaran dengan nilai yang dihasilkan oleh neuron. Minimalisasi nilai error dilakukan dengan cara mengubah nilai bobot di setiap neuron. Bobot diubah beberapa iterasi hingga menjadi konvergen, jumlah iterasi tersebut dinamakan dengan *epoch* (Wibawa, 2017). 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 :

Dimana :

: 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, dimana 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 3. 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* :

Dimana :

: 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, 2017)

Gambar 4. Ilustrasi Tanh

## 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).

*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 ekstrasi fitur dari setiap data supaya bisa membedakan suaru label dengan label yang lain. Pada sesi testing data-data yang diuji dapat dianalisa dari hasil sesi training (Azizah, Umayah, & Fajar, 2018).



(Sumber : Savalia & Emamian, 2018)

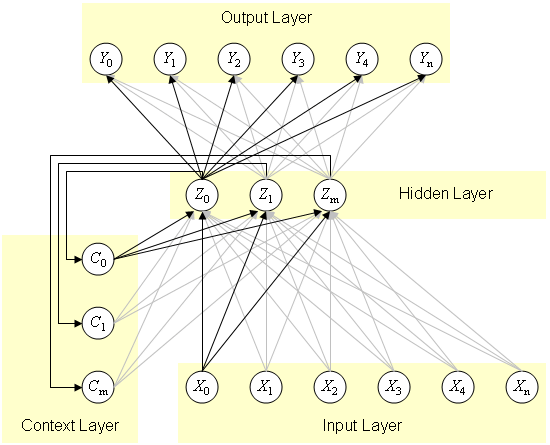
Gambar 5. Perbedaan Simple Neural Network dan Deep Learning

Pada *Neural Network* diilhami oleh pemrosesan informasi dan distribusi node komunikasi dalam sistem biologis dan terinspirasi dari cara kerja otak manusia karena terbukti Otak manusia melakukan sesuatu hal yang sama persis seperti, hirarki pertama pada neuron menerima formasi pada visual cortex yang sensitif terhadap gambaran tepi dan gumpalan khusus (François, 2018).

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).

## Recurrent Neural Network

*Recurrent Neural Network (RNN)* adalah model yang efisien untuk analisis sentimen. *RNN* menggunakan sel memori yang mampu menangkap informasi tentang urutan panjang (Patel & Tiwari, 2019). Dalam melakukan prediksi, RNN menggunakan data input pada waktu saat ini dan juga input dari data sebelumnya. Hubungan antar input tersebut dapat berguna untuk memberikan informasi kepada seluruh hidden layer. Sehingga dapat dikatakan bahwa RNN memiliki memori yang berisikan hasil rekaman informasi yang dihasilkan sebelumnya (Gulli & Pal, 2017).



(Sumber : Juanda, Jondri, & Rohmawati, 2018)

Gambar 6. Arsitektur Recurrent Neural Network

*RNN* merupakan variasi dari *Artificial Neural Network*, perbedaan utama yang terdapat pada model ini adalah sinyal dapat mengalir secara *forward* dan *backward* secara berulang. Untuk bisa melakukan hal tersebut, maka ditambahkan sebuah layer baru yang disebut dengan *context layer*. Selain melewati input antar layer, output dari setiap layer juga menuju ke context layer untuk digunakan sebagai inputan pada *timestep* berikutnya. RNN menyimpan informasi di *context layer*, yang membuatnya dapat mempelajari urutan data dan menghasilkan output atau urutan lain. Jika ditarik kesimpulan maka dapat dikatakan bahwa RNN memiliki memori yang berisikan hasil rekaman informasi yang dihasilkan sebelumnya (Juanda, Jondri, & Rohmawati, 2018). Pada layer tersembunyi *(hidden layer)* bertindak sebagai penyimpanan internal untuk menyimpan informasi yang dikumpulkan selama tahap awal pemrosesan data sekuensial. Alasan *RNN* disebut "berulang" adalah karena model melakukan tugas yang sama untuk setiap elemen urutan saat memanfaatkan informasi itu (Ludwig, 2019).

Diagram, schematic

Description automatically generated

(Sumber : Juanda, Jondri, & Rohmawati, 2018)

Gambar 7. Proses Recurrent Neural Network

Pada **Gambar 7** menunjukkan RNN pada posisi yang tidak dibuka ke full network. Dengan membuka gulungan RNN maka simpelnya kita menuliskan seluruh jaringan dengan urutan (sequence) secara lengkap (Juanda, Jondri, & Rohmawati, 2018). Berikut ialah keterangan simbol yang ada pada **Gambar 7** :

1. ialah input pada setiap *time step* biasanya berupa input sebuah vektor.
2. adalah “memory” pada *RNN* akan dihitung berdasarkan *hidden state* sebelumnya dan input pada langkah *time step* sehingga metode ini disebut sekuensial pada setiap
3. adalah output untuk setiap step

Pemetaan satu simpul S𝑡 dan output 𝑂𝑡 dapat ditulis sebagai (Tian, Ma, Zhang, & Zhan, 2018) :

Dimana dikatakan memori jaringan pada waktu , , , dan adalah matriks bobot berbagai di setiap layer, dan mewakili *input* dan *output* pada waktu , dan fungsi dan fungsi mewakili fungsi non-linear (Tian, Ma, Zhang, & Zhan, 2018).

Berbeda dengan koneksi bobot yang dibuat antara layer dalam *ANN*, *RNN* dapat menggunakan internal state (memori) untuk memproses urutan input (Che, Purushotham, Cho, Sontag, & Liu, 2018). *Hidden state* menyimpan informasi pada proses sebelumnya, dan *outputnya* diturunkan dari proses sekarang dengan memory sebelumnya. Secara teori, *RNN* juga mampu menangani dengan ketergantungan jangka panjang. Namun, dalam beberapa hal, *RNN* tidak dapat menghafal informasi sebelumnya dengan baik ketika interval waktu sangat panjang, masalah ini biasanya di sebut dengan *gradient vanishing problem* (Tian, Ma, Zhang, & Zhan, 2018). Dikarenakan RNN hanya mengingat beberapa langkah awal dalam urutan data sehingga tidak cocok untuk mengingat urutan yang lebih panjang. Untuk mengatasi kekurangan ini dan meningkatkan kinerja *RNN*, maka dikembangkanlah jenis jaringan arsitektur *RNN* khusus berulang lainnya yaitu *LSTM (Long Short-Term Memory)* (Moghar & Hamiche, 2020; Tian, Ma, Zhang, & Zhan, 2018).

## 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 8. Perulangan modul dalam 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 9. Alur Informasi Cell State pada LSTM

Keterangan (Pardede & Ibrahim, 2020) :

: *Cell state*

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

Pada **Gambar 9**, 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, 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).

Graphical user interface, application

Description automatically generated

(Sumber : Manu, 2021)

Gambar 10. Layer sigmoid mengeluarkan angka antara nol dan satu.

Pada **Gambar 10** 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)*.* Berikut ialah keterangan setiap gate yang ada pada **Gambar 8** :

### *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. Dimana 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 11** :

Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 11. Alur Informasi 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 11** 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 9**. *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 vector nilai baru yang akan disimpan pada memory cell (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 12** :

Diagram

Description automatically generated

(Sumber : Manu, 2021)

Gambar 12. Alur Informasi 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 12** 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. Dimana nilai ini didapatkan dari menggabungkan nilai yang terdapat pada forget gate dan input gate (Aldi, Jondri, & Aditsania, 2018). Dengan rumus seperti pada **Gambar 13** :

Diagram

Description automatically generated with medium confidence

(Sumber : Manu, 2021)

Gambar 13. Alur 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 13**, 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 () dimana LSTM akan mengalikan *cell state* lama dengan () kemudian ditambahkan dengan () . Ini adalah nilai kandidat baru, yang diskalakan berdasarkan seberapa banyak kami memutuskan untuk memperbarui setiap nilai negara bagian. (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 14** :

A picture containing text, clock

Description automatically generated

(Sumber : Manu, 2021)

Gambar 14. Alur Informasi 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 14** adalah tahap dimana perlu memutuskan apa yang akan hasilkan. *Output* akan didasarkan pada *cell state*, tetapi akan menjadi versi yang difilter. Pertama, kita menjalankan layer sigmoid yang memutuskan bagian mana dari *cell state* yang akan kita 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).

## Optimasi Adaptive Moment Estimation (ADAM)

Adaptive Moment Estimation atau yang biasa disingkat Adam adalah sebuah metode optimasi. Adam melakukan penyimpanan nilai Gradient-Gradient di perhitungan sebelumnya untuk menghitung rata-rata nilai Gradient pada momentum pertama maupun momentum kedua. Dari kedua rata-rata tersebut di adaptasi parameter -nya untuk digunakan sebagai pembaharuan nilai bobot (Winoto, 2020).

Adam menyimpan rata-rata gradien proses sebelumnya secara eksponensial. Nilai learning rate standar pada Adam adalah 0,001. Rumus perhitungan optimasi Adam ditunjukkan pada persamaan berikut (Miranda, Novamizanti, & Rizal, 2020) :

Dimana (Miranda, Novamizanti, & Rizal, 2020) :

: adalah parameter hasil pembaruan

: adalah parameter hasil pembaruan sebelumnya

: adalah *learning rate*

: adalah gradien kuadrat momen orde pertama

: adalah gradien kuadrat momen orde kedua

: merupakan scalar kecil untuk mencegah pembagian dengan 0

## 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 :

Dimana (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 biasa digunakan sebagai indikator untuk mengukur dan membandingkan kemiripan hasil prediksi dan data asli (Aprian, Azhar, & Nastiti, 2020).

## Batch Size dan Epoch

*Epoch* adalah ketika seluruh dataset 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 kedalam proses training sehingga akan membutuhkan waktu cukup lama. Agar mempermudah dan mempercepat proses training, biasanya 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 dataset yang besar. Penentuan nilai dari batch size biasanya tergantung peneliti dengan melihat banyak sampel (Thohari & Hertantyo, 2018).

# DAFTAR PUTAKA

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

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/

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

Che, Z., Purushotham, S., Cho, K., Sontag, D., & Liu, Y. (2018, March 26). Recurrent Neural Networks for Multivariate Time Series with Missing Values. *Scientific reports, 8*, 1-12. doi:10.1038/s41598-018-24271-9

Damjan, P., & Vatanawood, W. (2017, March 17). Translating UML State Machine Diagram into Promela. *Proceedings of the International MultiConference of Engineers and Computer Scientists, 1*. Diambil kembali dari http://www.iaeng.org/publication/IMECS2017/IMECS2017\_pp512-516.pdf

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

Dermawan, J., & Hartini, S. (2017, September 2). IMPLEMENTASI MODEL WATERFALL PADA PENGEMBANGAN SISTEM INFORMASI PERHITUNGAN NILAI MATA PELAJARAN BERBASIS WEB PADA SEKOLAH DASAR AL-AZHAR SYIFA BUDI JATIBENING. *Paradigma, 19*. doi:10.31294/p.v19i2.2131

Flywind. (2017, 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

Gulli, A., & Pal, S. (2017). *Deep Learning with Keras.* Packt Publishing Ltd. Diambil kembali dari https://books.google.co.id/books?id=20EwDwAAQBAJ

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.

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

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

Ludwig, S. A. (2019, October 10). Comparison of Time Series Approaches applied to Greenhouse Gas Analysis: ANFIS, RNN, and LSTM. *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. doi:10.1109/FUZZ-IEEE.2019.8859013

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

Miranda, N. D., Novamizanti, L., & Rizal, S. (2020, August 3). CONVOLUTIONAL NEURAL NETWORKPADA KLASIFIKASI SIDIK JARI MENGGUNAKAN RESNET-50. *Jurnal Teknik Informatika (JUTIF)*, 61-68. doi:10.20884/1.jutif.2020.1.2.18

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

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

Patel, A., & Tiwari, A. K. (2019, February 8). Sentiment Analysis by using Recurrent Neural Network. *Proceedings of 2nd International Conference on Advanced Computing and Software Engineering (ICACSE) 2019*, 4. doi:10.2139/ssrn.3349572

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

Ramadhan, R. F., & Mukhaiyar, R. (2020, November 8). Penggunaan Database Mysql dengan Interface PhpMyAdmin sebagai Pengontrolan Smarthome Berbasis Raspberry Pi. *Jurnal Teknik Elektro Indonesia*, 129-134. doi:10.24036/jtein.v1i2.55

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

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

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

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

Supriatna, A. D., & Sherin, N. (2017, October 31). Perancangan Sistem Informasi Penjualan Mobil di Dealer Lung Ma Motor. *Jurnal Algoritma*, 305-313. doi:https://doi.org/10.33364/algoritma/v.14-2.305

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.

Syahrudin, A. N., & Kurniawan, T. (2018, Juni). INPUT DAN OUTPUT PADA BAHASA PEMROGRAMAN PYTHON. *Jurnal Dasar Pemograman Python STMIK*, 1-7. Diambil kembali dari https://www.researchgate.net/profile/Tedi-Kurniawan-2/publication/338385483\_INPUT\_DAN\_OUTPUT\_PADA\_BAHASA\_PEMROGRAMAN\_PYTHON/links/5e10643392851c8364b029c3/INPUT-DAN-OUTPUT-PADA-BAHASA-PEMROGRAMAN-PYTHON.pdf

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

Tian, C., Ma, J., Zhang, C., & Zhan, P. (2018, December 12). A Deep Neural Network Model for Short-Term Load Forecast Based on Long Short-Term Memory Network and Convolutional Neural Network. *Energies*, 3-4. doi:10.3390/en11123493

Wali, M. (2020). *Modul Praktikum Rekayasa Perangkat Lunak.* (A. D. Rahma, Rosalita, & H. N. Putra, Penyunt.) Bandung, Jawa Barat, Indonesia: Ellunar.

Wibawa, M. S. (2017, May 24). Pengaruh Fungsi Aktivasi, Optimisasi dan Jumlah Epoch Terhadap Performa Jaringan Saraf Tiruan. *JURNAL SISTEM DAN INFORMATIKA, 11*, 167-174. Diambil kembali dari https://www.jsi.stikom-bali.ac.id/index.php/jsi/article/view/129

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.

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