

## BAB II

### KAJIAN LITERATUR

#### 2.1 Polusi Udara

Pencemaran udara menurut Peraturan Pemerintah Republik Indonesia Nomor 41 Tahun 1999 tentang pengendalian pencemaran udara adalah masuknya atau dimasukkannya zat, energi, dan atau komponen lain ke dalam udara oleh kegiatan manusia, sehingga mutu udara turun sampai ke tingkat tertentu yang menyebabkan udara tidak dapat memenuhi fungsinya [11]. Pencemaran udara merupakan masalah yang sering terjadi karena faktor perekonomian yang semakin berkembang. Berkembangnya kehidupan ekonomi membuat masyarakat tertarik untuk menggunakan alat-alat berteknologi tinggi sehingga menimbulkan pencemaran udara yang berasal dari industri pabrik, mobil, motor dan lain-lain. Pada tahun 2010, sekitar 3,3 juta orang di seluruh dunia meninggal hanya dikarenakan menghirup debu-debu kecil yang beterbangan di udara dan diperkirakan akan berlipat ganda pada tahun 2050 [12]. Hal ini terjadi karena polutan yang dikeluarkan dari asap industri, asap transportasi serta polutan yang berasal dari gunung meletus dan kebakaran hutan membuat menurunnya kualitas udara yang mengakibatkan dampak negatif bagi kehidupan manusia, hewan dan tumbuh-tumbuhan.

#### 2.2 Forecasting

Peramalan (*forecasting*) adalah seni dan ilmu untuk memperkirakan kejadian di masa depan [13]. Peramalan bertujuan untuk mendapatkan peramalan yang meminimalkan kesalahan pada masa depan untuk mendapatkan hasil yang terbaik. Dapat disimpulkan bahwa pengertian peramalan adalah peristiwa yang belum terjadi dengan tujuan untuk memperkirakan peristiwa-peristiwa yang akan terjadi dimasa depan dengan memperhatikan data-data dari masa lalu. Dengan cara membentuk model matematis atau prediksi intuisi bersifat subyektif yang disesuaikan dengan pertimbangan yang baik [13]. Standar tahapan yang sudah disepakati dalam menerapkan model peramalan waktu diantaranya adalah identifikasi tujuan peramalan, penentuan periode waktu peramalan, pemilihan model peramalan, persiapan data (*data cleaning*), penerapan model peramalan, analisis hasil peramalan dan evaluasi hasil peramalan [14]. *Forecasting* terbagi menjadi 3 prakiraan jangka waktu yaitu [15]:

1. Prediksi Jangka Pendek (*Short-Term Forecast*)

Prediksi jangka pendek adalah prediksi untuk jangka waktu hari, mingguan dan bulanan.

2. Prediksi Jangka Menengah (*Medium-Term Forecast*)

Prediksi jangka menengah adalah prediksi untuk jangka waktu 1 tahun hingga 2 tahun ke depan.

3. Prediksi Jangka Panjang (*Long-Term Forecast*)

Prediksi jangka panjang adalah prediksi untuk jangka waktu yang sangat lama hingga 5 atau 10 tahun ke depan.

### 2.3 Data Deret Waktu

Data deret waktu (*time series*) adalah urutan pengamatan berdasarkan urutan waktu karakteristik kuantitatif dari suatu peristiwa atau serangkaian peristiwa dalam jangka waktu tertentu [13]. Data *Time Series* adalah data yang dikumpulkan dari waktu ke waktu untuk memberikan gambaran perkembangan kegiatan dari waktu ke waktu. Analisis deret waktu memungkinkan untuk memahami bagaimana satu atau lebih peristiwa terjadi dan bagaimana hubungannya dengan peristiwa lain. Data deret waktu sudah banyak diterapkan, seperti pada grafik saham, prakiraan cuaca atau pertumbuhan populasi. Model *time series* merupakan peramalan kuantitatif berdasarkan analisis pola hubungan antara variabel yang akan dicari (*dependency*) dan variabel yang mempengaruhinya (*independence*) dan berubah dari waktu ke waktu seperti minggu, bulan, triwulan, semester, tahun dan lain-lain. Data masa lalu sangat penting dalam memprediksi kondisi di masa yang akan datang karena dari penelitian-penelitian yang sudah pernah dilakukan, data masa lalu ternyata tidak sepenuhnya mempunyai pola gerakan yang acak. Pergerakan data masa lalu dari waktu ke waktu mempunyai ciri-ciri tertentu yang biasa dijadikan sebagai dasar dalam melakukan analisis prediksi. Sampai batas-batas tertentu data masa lalu dari waktu ke waktu seakan-akan menunjukkan pola yang pasti [16].

Tujuan dari pendekatan ini adalah untuk menemukan pola pada deret masa lalu dan membuat inferensi pola-pola tersebut di masa depan sehingga memungkinkan hasilnya digunakan sebagai referensi atau acuan untuk memprediksi nilai masa depan [16]. Contoh *time series* sederhana dapat dilihat seperti Gambar 2.1 di bawah ini:

Timestamp	Value
17/12/2021 09:05:39	3
17/12/2021 09:05:39	6
17/12/2021 09:34:27	7
17/12/2021 09:34:27	8
17/12/2021 09:59:02	4
17/12/2021 11:37:41	5

Gambar 2.1 Data deret waktu sederhana

Di mana unsur-unsurnya adalah:

1. *Timestamp* adalah sebuah tanda waktu pada kejadian yang didaftarkan, yang nanti akurasiya akan tergantung pada peristiwa yang diukur.
2. *Value* adalah nilai fenomena pada masa lalu, dapat bernilai hanya satu atau lebih nilai. Apabila lebih dari satu nilai per *timestamp* berarti memiliki deret waktu *multivariabel*.

Jenis-jenis *time series* [17]:

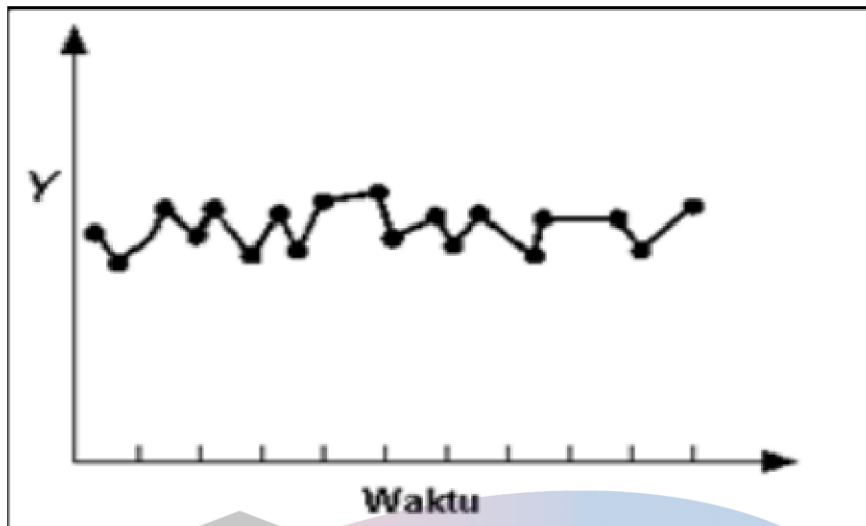
1. *Reguler time series* adalah data yang memiliki catatan pada setiap periode waktu yang sama misalnya suhu per hari yang dapat diasumsikan sebagai sebuah rangkaian biasa karena satu catatan per hari, satu catatan per unit waktu.
2. *Irreguler time series* berbeda dengan *reguler time series*, *irreguler time series* adalah data yang dikumpulkan tanpa adanya frekuensi yang ditentukan, contohnya adalah sebuah *log* akses ke sebuah situs web. Karena setiap *user* tidak melakukan akses secara bersamaan setiap waktu. Dan hal ini membuat catatan dengan interval waktu yang berbeda-beda.

### 2.3.1 Pola Runtun Waktu

Secara umum terdapat empat pola dalam pola runtun waktu yaitu horizontal, *trend*, musiman dan siklus [18].

1. Horizontal

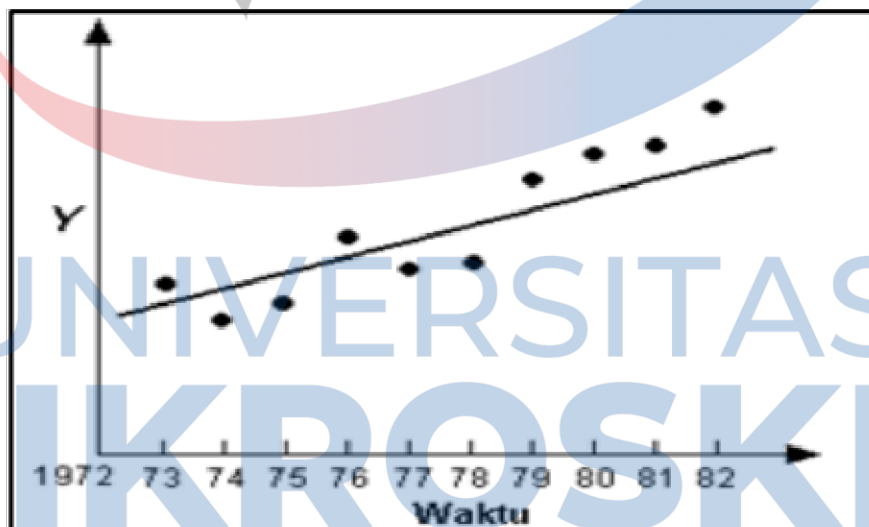
Pola data horizontal terjadi ketika data yang diobservasi berubah-ubah di sekitar tingkatan rata-rata yang konstan.



Gambar 2.2 Pola Data Horizontal [19]

2. Tren (*Trend*)

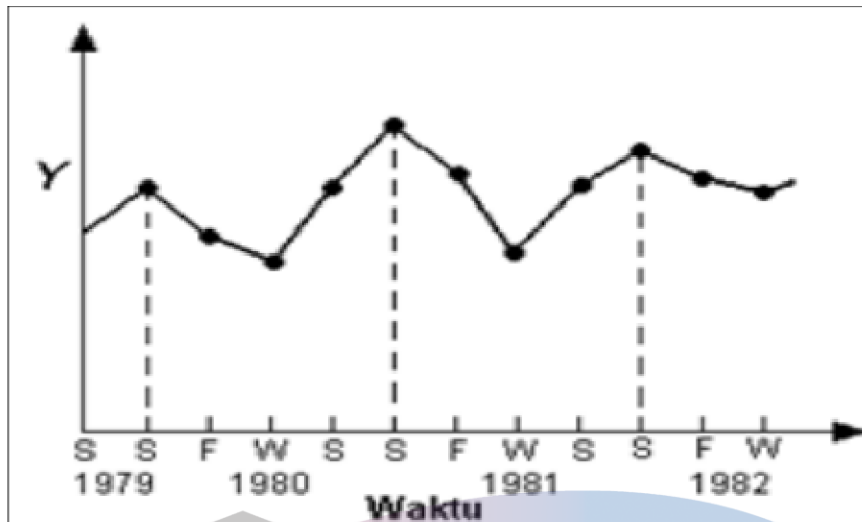
Pola data tren dapat dilihat dengan adanya sebuah peningkatan maupun penurunan dalam jangka panjang pada sebuah data yang dapat disebut dengan istilah “perubahan arah”.



Gambar 2.3 Pola Data Tren [19]

3. Musiman (*Seasonal*)

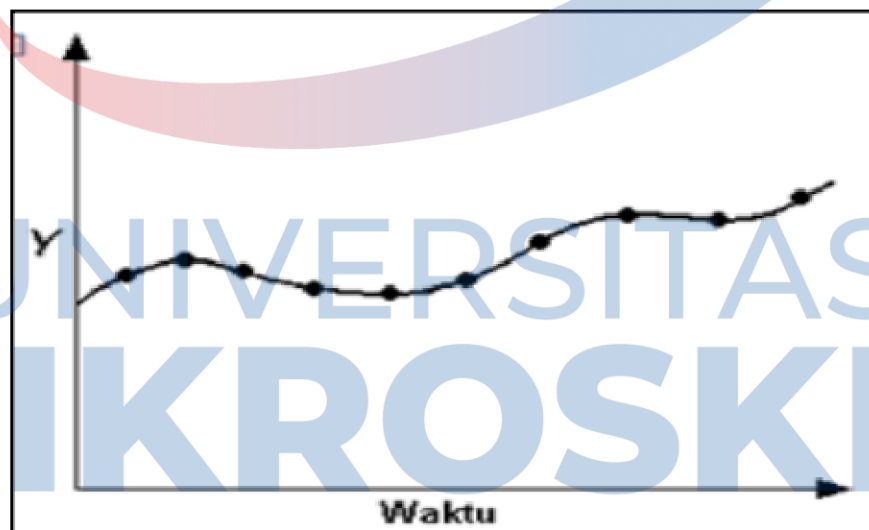
Pola data musiman dapat terjadi karena perubahan rangkaian waktu yang dipengaruhi oleh musiman seperti hari-hari tertentu dalam satu waktu atau kejadian yang merupakan tetap.



Gambar 2.4 Pola Data Musiman [19]

#### 4. Siklus (*Cyclic*)

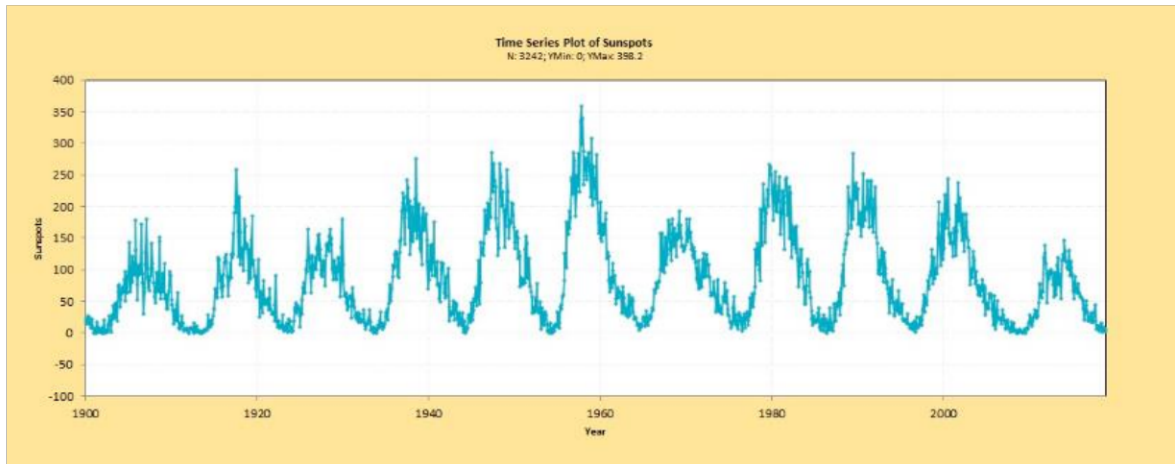
Pola data siklus dapat terjadi dengan adanya naik turun dalam sebuah data tetapi bukan dari frekuensi yang tetap. Pola siklus biasanya dipengaruhi oleh bisnis dan ekonomi.



Gambar 2.5 Pola Data Siklus [20]

### 2.3.2 Plot Runtun Waktu

Plot adalah langkah-langkah yang digunakan untuk menunjukkan kumpulan-kumpulan runtun waktu yang jelas di mana ada hubungan antara sebuah variabel dengan variabel yang lain dapat dilihat melalui grafik, *chart* dengan plot (alur) dan membantu membaca nilai pada variabel seperti Gambar 2.6 di bawah ini [21].



Gambar 2.6 Plot runtun waktu bintang matahari [21]

Berikut jenis dari plot dalam deret waktu (*time series*) [22]:

1. *Plot Lollipop*

*Plot Lollipop* atau yang sering disebut *barplot* dimana bentuk plot diubah ke dalam bentuk garis dan titik-titik yang menunjukkan hubungan antara variabel numerik dan variabel kategorik. Pada plot ini menampilkan hierarki dengan warna yang dapat dibedakan berdasarkan populasi atau grup.

2. *Plot Treemap*

*Plot Treemap* adalah sebuah teknik membuat visualisasi data yang berfungsi untuk menampilkan data hierarki dengan persegi bersarang yang mewakili data seperti berbentuk pohon. Data disusun dengan cabang pembantu dan direpresentasikan menggunakan persegi panjang, dimensi serta warna perseginya dihitung dengan variabel kuantitatif dan setiap persegi mewakili dua nilai numerik.

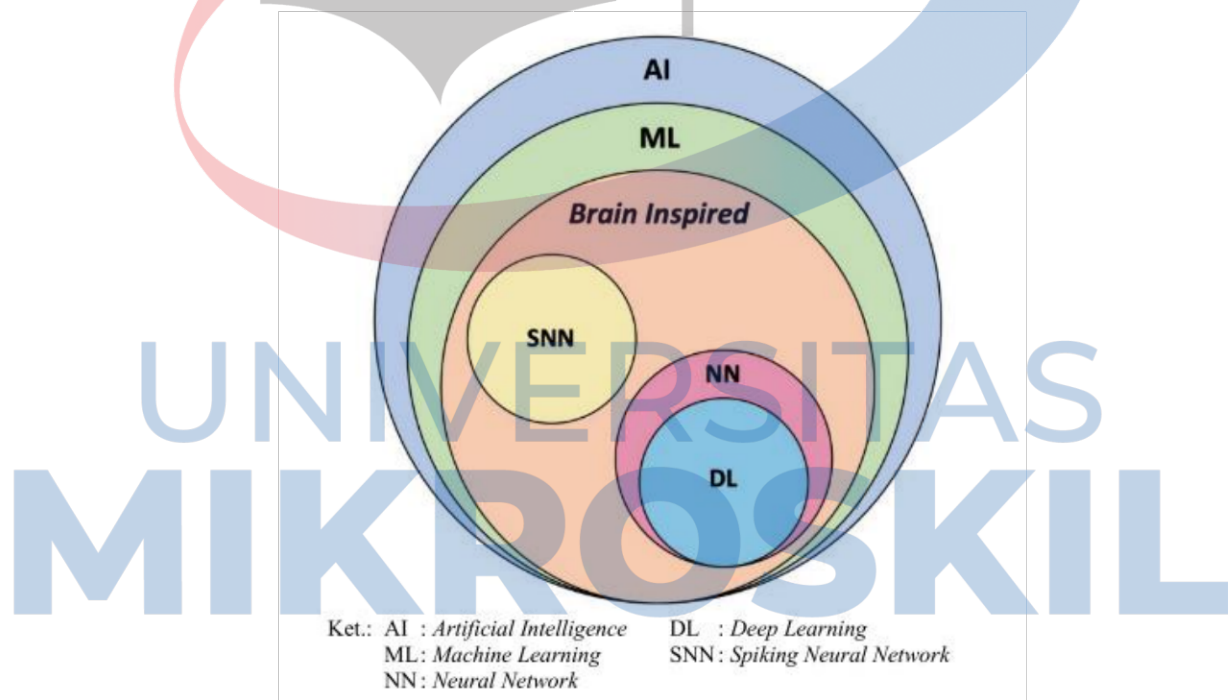
## 2.4 Analisis Time Series

Analisis *Time Series* merupakan analisis yang dilakukan berdasarkan sebuah data atau pengamatan yang mengarah pada waktu atau kronologis pada sebuah variabel yang diamati [23]. Analisis *Time Series* sangat berguna pada data yang perubahannya dipengaruhi oleh sebuah waktu atau pengamatan terdahulu. Dalam pengembangannya, Analisis *Time Series* digunakan di beberapa bidang seperti ekonomi, keuangan, transportasi, kesehatan dan masih banyak lagi. Untuk menentukan model yang cocok untuk meramal sebuah data terdapat beberapa proses dalam analisis *time series* yaitu: Stasioneritas data, *Parameter Estimation*, *Model Specification*, *Model Checking*, *Unit Root Test* dan *Forecasting*. Tujuan dari analisis *time series* adalah untuk menemukan sebuah pola dalam deret data historis lalu menekstrapolasikan pola tersebut ke masa depan [24].

Dalam melakukan analisis deret waktu tentu sangat diperlukan sebuah perangkat lunak atau *software* seperti *R Programming*, *Anaconda Navigator*, *Minitab* dan *SAS* dimana nantinya *software* tersebut akan bekerja bersamaan dengan algoritma, teknik serta model prediksi yang akan digunakan. Biasanya *R* lebih sering digunakan dalam melakukan analisis deret waktu dikarenakan memiliki kemampuan yang baik dalam melakukan analisis deret waktu dan menghasilkan perkiraan. *R* juga tergolong sangat efektif dalam pengelolaan data, selain itu *R* juga bersifat *open source* atau tidak dibatasi [25].

## 2.5 Deep Learning

Sejak tahun 1950-an, cabang *Artificial Intelligence* yang disebut dengan *Machine Learning* telah berkembang pesat dan diimplementasikan di beberapa bidang. *Neural Network* merupakan implementasi dari *Machine Learning*, sedangkan *Deep Learning* merupakan implementasi dari *Neural Network* [26]. Taksonomi ini dapat dilihat pada Gambar 2.7



Gambar 2.7 Taksonomi AI [27]

*Deep Learning* yang populer sejak tahun 2006 ini menggunakan mekanisme *Deep architecture learning* atau metode pembelajaran hierarkis. Pembelajaran dalam konteks ini adalah proses yang melibatkan proses estimasi parameter model sehingga model (algoritma) yang dikembangkan dapat menyelesaikan tugas atau masalah tertentu [27].

*Deep Learning* menggunakan beberapa lapisan (*layer*) diantara lapisan masukan (*input layer*) dan lapisan keluaran (*output layer*). Arsitektur ini dapat digunakan untuk melakukan pemrosesan nonlinier dengan beberapa tahapan, yang hasilnya dapat digunakan untuk pembelajaran fitur dan klasifikasi pola. Berbagai lapisan dalam DL dapat digunakan untuk melakukan berbagai tingkat abstraksi yang berbeda-beda [28].

*Deep Learning* diartikan sebagai sebuah kelas dalam algoritme ML yang menggunakan beberapa lapisan pemrosesan nonlinier yang diatur secara cascade untuk ekstraksi fitur (*feature extraction*) dan *transformation*. Setiap lapisan menggunakan *output* dari lapisan sebelumnya sebagai *input*-nya. Algoritma yang digunakan dapat bertipe *supervised* dan *unsupervised* serta implementasinya dapat digunakan sebagai *pattern analysis (unsupervised)* dan klasifikasi (*supervised*) [26].



Gambar 2.8 Kategori dalam Deep Learning [27]

Beberapa teknik dalam *Deep Learning* dapat dikategorikan menjadi *supervised*, *semi-supervised*, dan *unsupervised*. Kategori lain, seperti *Reinforcement Learning* (RL) atau *Deep RL* (DRL) seringkali dikategorikan menjadi *semi-supervised* atau *unsupervised*. Kategori tersebut dapat dilihat pada Gambar 2.7 diatas dan diartikan sebagai berikut [27]:

1. *Deep Supervised Learning*: Teknik learning yang digunakan dalam kategori ini menggunakan data yang telah diberi label sebelumnya (*labeled data*).

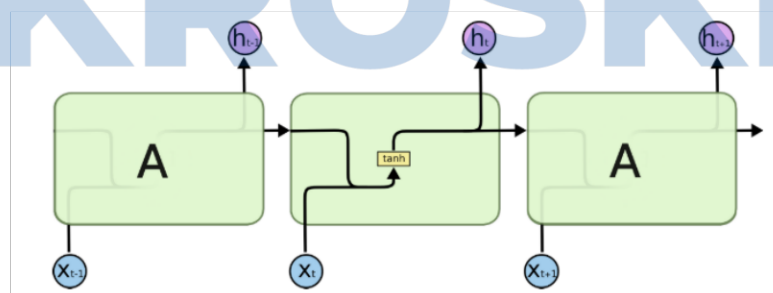


Contoh: *Deep Neural Networks* (DNN), CNN, (RNN), termasuk LSTM dan *Gated Recurrent Unit* (GRU).

2. *Deep Semi-Supervised Learning*: *Semi-supervised learning* menggunakan teknik learning yang menggunakan sebagian data yang telah diberi label sebelumnya (*partially labeled data*). Contoh: DRL, *Generative Adversarial Networks* (GAN), Serta RNN, termasuk LSTM dan GRU.
3. *Deep Unsupervised Learning*: Teknik learning ini menggunakan data yang tidak diberi label sebelumnya (*unlabeled data*). *Auto Encoders* (AE), *Restricted Boltzmann Machines* (RBM), dan generasi terbaru dari GAN.
4. *Deep Reinforcement Learning*: Teknik learning ini digunakan pada lingkungan atau environments yang tidak diketahui (*unknown environments*). Pada tahun 2013, DRL dimulai dengan hadirnya *Google DeepMind*.

## 2.6 Long Short-Term Memory

Model *Long Short-Term Memory* (LSTM) merupakan evolusi dari arsitektur *Recurrent Neural Network* (RNN) yang biasa digunakan pada masalah-masalah yang berkaitan dengan *deep learning*, dengan melakukan modifikasi pada RNN menambahkan *memory cell* yang dapat menyimpan informasi untuk jangka waktu yang lama. RNN adalah algoritma yang sangat kuat yang dapat mengklasifikasikan, mengelompokkan dan membuat prediksi tentang data, terutama deret waktu dan teks [6]. *Long Short-Term Memory* (LSTM) merupakan sebuah evolusi dari arsitektur RNN, yang diperkenalkan pertama kali oleh Hochreiter & Schmidhuber pada tahun 1997, kemudian dikembangkan dan dipopulerkan oleh banyak periset. Seperti RNN, jaringan LSTM (*LSTM Network*) juga terdiri dari modul-modul dengan pemrosesan berulang, seperti yang ditunjukkan pada Gambar 2.9



Gambar 2.9 Looping pada RNN dengan satu layer [29].

Pada Gambar 2.9 di atas adalah looping pada RNN yang terdiri dari satu layer yakni *tanh layer* yang akan dijabarkan pada persamaan berikut:

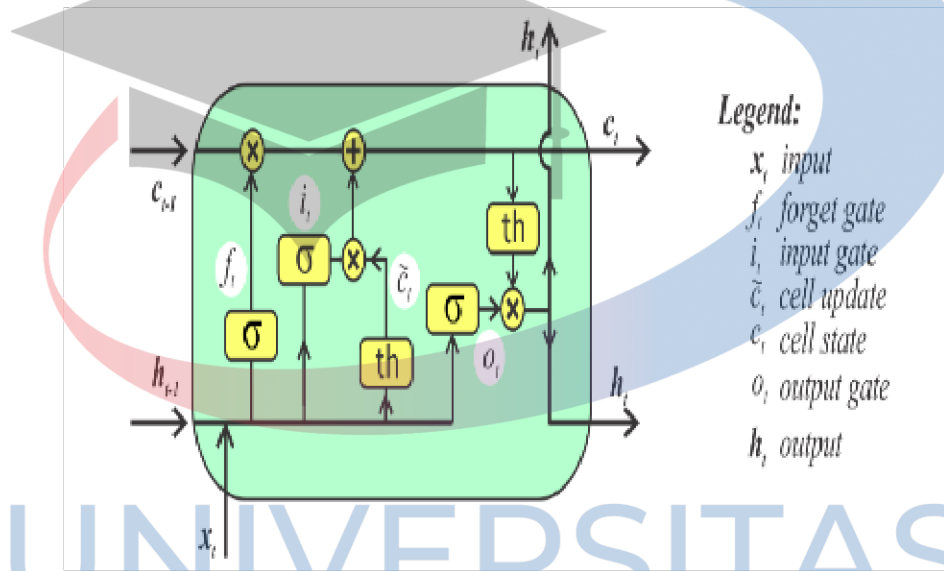
$$\tanh(x) = 2\sigma(2x) - 1 \quad (1)$$

dimana:

$\sigma$  : fungsi sigmoid

$x$  : data input

Arsitektur umum LSTM terdiri dari *memory cell*, *input gate*, *output gate* dan *forget gate*. LSTM *cell* mengambil masukan dan menyimpannya untuk beberapa waktu. Secara intuitif, *input gate* mengontrol sejauh mana nilai tetap di dalam *cell* dan *output gate* mengontrol sejauh mana nilai dalam *cell* digunakan untuk menghitung aktivasi keluaran dari unit LSTM. Adapun arsitektur LSTM akan dijelaskan pada Gambar 2.10.



Gambar 2.10 Arsitektur LSTM [30].

Pada Gambar 2.10 di atas, didapatkan rumus yang diuraikan sebagai berikut:

$$f_t: \sigma(W_f \cdot, h_{t-1}, x_{t-} + b_f)$$

$$i_t: \sigma(W_i \cdot, h_{t-1}, x_{t-} + b_i)$$

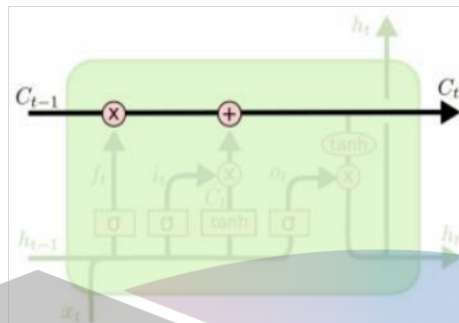
$$c_t : f_t * C_{t-1} + i_t \bar{c}_t$$

$$o_t : \sigma(W_o \cdot, h_{t-1}, x_{t-} + b_o) \quad (2)$$

$$h_t: \tanh(c_t)$$

$$\bar{c}_t : \tanh(W_c \cdot, h_{t-1}, x_{t-} + b_c)$$

Langkah-langkah LSTM seperti Gambar 2.11 akan diuraikan satu persatu pada bagian ini *Cell State* adalah kunci utama di dalam metode LSTM. *Cell state* adalah garis horizontal yang melewati bagian atas diagram yang menghubungkan semua *output layer* pada LSTM seperti Gambar 2.10



Gambar 2.11 Cell state pada LSTM [29].

LSTM memiliki *gates* yang terdiri dari *sigmoid layer* dan *pointwise operation*. Dimana *gates* ini dapat menambah ataupun menghapus informasi yang mana informasi tersebut akan diteruskan atau diberhentikan. *Sigmoid layer* ini akan memadatkan hasil *output* menjadi *range* [0,1]. Angka 0 menunjukkan informasi akan diberhentikan sedangkan angka 1 menunjukkan akan diteruskan.

Persamaan sigmoid diuraikan pada Persamaan di bawah ini:

$$\sigma(x) = \frac{1}{(1 + g^{-x})} \quad (3)$$

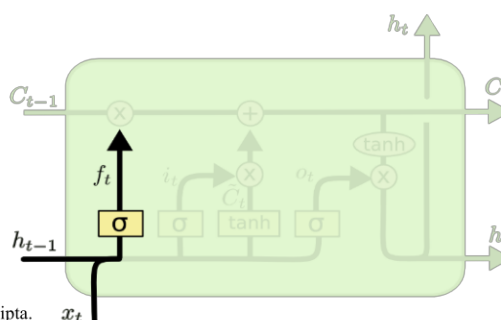
dimana:

x : data input

c : konstanta matematika (2,71828 18284 59405 23536 02784 71352)

Berikut tahapan jalannya metode LSTM yakni sebagai berikut:

Tahap pertama pemrosesan metode LSTM yaitu mengambil keputusan data apa yang akan dihilangkan dari *cell state* ( $C_t$ ). Proses tersebut dilakukan oleh *layer sigmoid* ( $\sigma$ ) dengan nama *Layer Forget Gate*. *Layer sigmoid* tersebut memproses ( $h_{t-1}$ ) dan ( $X_t$ ) sebagai suatu input baru kemudian menghasilkan sebuah output (menjadi nilai diantara 0 dan 1) pada ( $C_{t-1}$ ) yang dapat dilihat pada Gambar 2.12.



Gambar 2.12 Layer Forget Gate [29].

Persamaan *forget gate* diuraikan pada persamaan berikut:

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

dimana:

$f_t$  : *forget gate*

$\sigma$  : *layer sigmoid*

$W_f$  : nilai *weight forget gate*

$h_{t-1}$  : nilai *output* sebelumnya

$x_t$  : nilai *input* baru

$b_f$  : nilai *bias forget gate*

Dimana nilai *weight* dan nilai *output* diuraikan pada persamaan berikut:

$$W = -\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}} \quad (5)$$

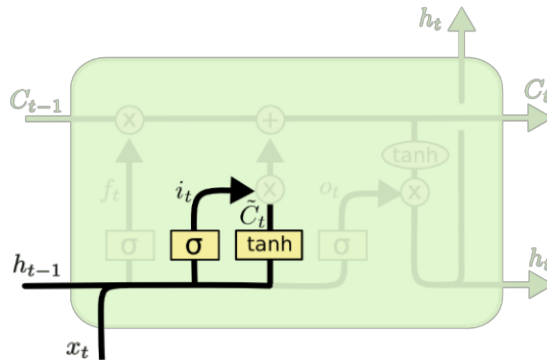
dimana:

$W$  : nilai *weight forget gate*

$d$  : jumlah data

$h_t$  : nilai *output* orde ke- $t$

Selanjutnya tahap kedua akan diputuskan data apa saja yang akan tersimpan pada *cell state*. Dalam tahap ini terdapat dua *layer gate* yaitu *input gate layer* dan *tanh layer*, layer pertama yaitu *input gate layer* akan memproses dan mengambil keputusan nilai yang akan diperbaharui dan menghasilkan  $i_t$ . Kemudian, *tanh layer* membuat nilai baru kemudian ditambahkan *cell state*. Langkah selanjutnya yaitu menggabungkan masing-masing hasil dari *layer input gate* dan *layer tanh*.



Gambar 2.13 Layer Input Gate dan Layer Tanh [29].

Langkah tersebut diuraikan pada persamaan berikut:

$$i_t = \sigma(W_i, h_{t-1}, x_{t-} + b_i) \quad (6)$$

dimana:

$i_t$  : *Input gate*

$W_i$  : nilai *weight input gate*

$b_i$  : nilai *bias input gate*

Persamaan untuk nilai baru diuraikan sebagai berikut:

$$\bar{C}_t = \tanh(W_c, h_{t-}, x_{t-} + b_c) \quad (7)$$

dimana:

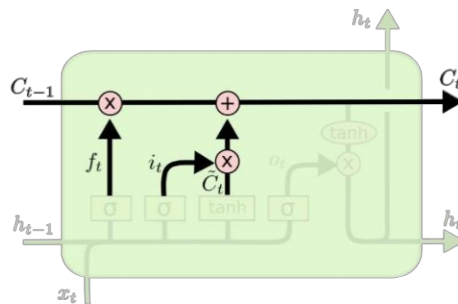
$\bar{C}_t$  : nilai baru yang ditambahkan pada *cell state*

$\tanh$  : fungsi dari *tanh*

$W_c$  : nilai *weight Cell State*

$b_c$  : nilai *bias Cell State*

Tahap ketiga yaitu akan dilakukan pembaharuan pada *cell state* lama  $C_{t-1}$ , menjadi *cell state* yang baru  $C_t$  yang dapat dilihat pada Gambar 2.14



Gambar 2.14 Layer New Cell State [29].

Dan persamaannya diuraikan sebagai berikut:

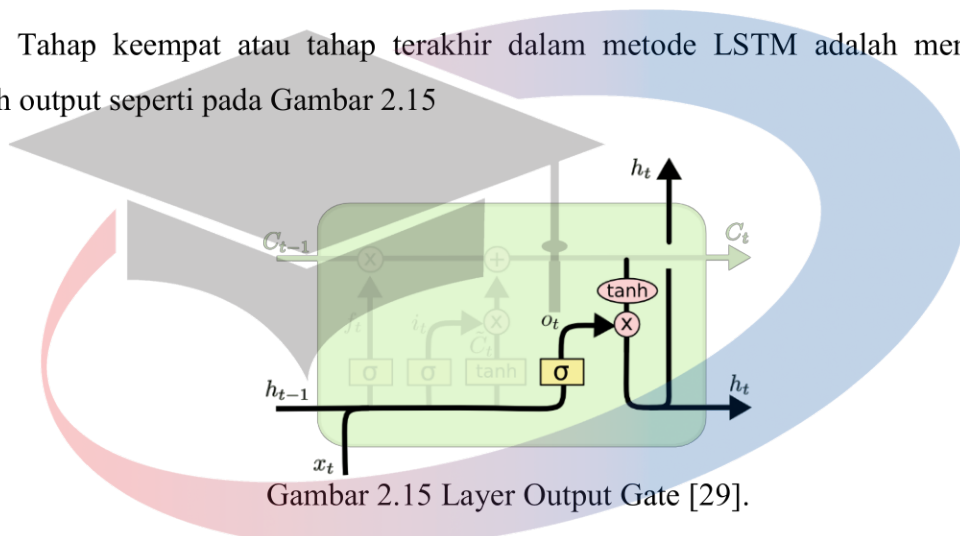
$$C_t = f_t C_{t-1} + i_t \bar{C}_t \quad (8)$$

dimana:

$C_t$  : *Cell State* baru

$C_{t-1}$  : *Cell State* lama

Tahap keempat atau tahap terakhir dalam metode LSTM adalah menghasilkan sebuah output seperti pada Gambar 2.15



Gambar 2.15 Layer Output Gate [29].

Pertama *layer sigmoid* ( $\sigma$ ) akan mengambil keputusan bagian mana dari *cell state* yang menjadi *output*. Kemudian *output* akan dikirim ke dalam *layer tanh* (dengan nilai antara -1 dan 1), lalu dikirim menuju *sigmoid gate* agar *output* yang dihasilkan sama dengan *output* sebelumnya.

$$o_t = \sigma(W_o, h_{t-1}, x_{t-1} + b_o) \quad (9)$$

dimana:

$o_t$  : *output gate*

$W_o$  : nilai *weight output gate*

$b_o$  : nilai bias *output gate*

Setelah didapatkan nilai dari output gate, kemudian menempatkan *cell state* melalui *tanh* untuk mendapatkan nilai dengan *range* -1 hingga 1. Kemudian mengalikannya dengan output gate dari *sigmoid layer*. Persamaan nilai output orde t dirumuskan pada Persamaan 10 berikut:

$$h_t = o_t * \tanh(c_t) \quad (10)$$

dimana:

$h_t$  : nilai *output* orde t

$o_t$  : *Output gate*

$C_t$  : *cell state*

## 2.7 Normalisasi Data

Normalisasi merupakan tahap kedua dari pra-proses. Pada tahap ini terjadi proses penskalaan, dimana data akan diubah dalam rentang 0 hingga 1 [31]. Tujuan dari normalisasi data adalah untuk menyetarakan *range* setiap variabel dengan variabel lain. Pentingnya normalisasi data sebelum pelatihan jaringan saraf untuk mempercepat perhitungan dan mendapatkan hasil yang baik dalam prediksi. Data *ISPU* akan dinormalisasi dengan menggunakan metode *Min-Max Normalization*.

Dengan demikian normalisasi dilakukan untuk menyamaratakan data. Normalisasi dapat dilakukan dengan rumus sebagai berikut:

$$X' = \frac{(x - x_{min})}{(x_{max} - x_{min})} \quad (11)$$

dimana:

$X'$  : data hasil normalisasi

$x$  : data asli

$x_{min}$  : nilai minimum dari data x

$x_{max}$  : nilai maximum dari data x

## 2.8 Denormalisasi

Denormalisasi adalah proses pengembalian data hasil normalisasi ke dalam data asli atau data sebenarnya [32]. Hal tersebut dilakukan guna melihat hasil prediksi dengan cara membandingkan dengan data sebenarnya. Denormalisasi dalam *range* [0,1] dinyatakan dengan rumus pada persamaan sebagai berikut:

$$x_t = x(X_{max} - X_{min}) + X_{min} \quad (12)$$

dimana:

$X_t$  : nilai dari data normalisasi

$x$  : hasil *output*

$X_{\min}$  : nilai minimum dari data aktual keseluruhan

$X_{\max}$  : nilai maximum dari data aktual keseluruhan

## 2.9 Evaluasi Model

Untuk mengukur performansi pada sebuah model dapat diuji dengan melakukan perbandingan antara nilai aktual dengan nilai prediksi. Beberapa matrik evaluasi model yaitu: *root mean square error* (RMSE), *mean absolute percentase error* (MAPE), *mean squared error* (MSE), *mean absolute error* (MAE), *mean absolute deviation* (MAD) yang berguna untuk mengevaluasi kinerja pada setiap model.

### 1. Root Mean Square Error (RMSE)

*Root Mean Square Error* adalah penjumlahan nilai kuadrat error atau selisih antara nilai aktual dengan nilai prediksi, kemudian membagi jumlah nilai tersebut dengan banyaknya waktu dari sebuah data peramalan atau dapat dirumuskan pada persamaan sebagai berikut [33]:

$$RMSE = \sqrt{\sum \frac{(Aktual - Prediksi)^2}{n}} \quad (13)$$

dimana:

Aktual : Nilai aktual

Prediksi : Nilai Prediksi

$n$  : Jumlah data

### 2. Mean Absolute Percentase Error (MAPE)

*Mean Absolute Percentase Error* adalah nilai rata-rata absolut antara nilai aktual dan nilai prediksi yang dapat disebut sebagai hasil persenan dari nilai prediksi. Penggunaan MAPE pada evaluasi dari hasil aktual dapat melihat tingkat akurasi terhadap nilai peramalan dan nilai aktual [34]. MAPE dapat dirumuskan pada persamaan sebagai berikut:



$$MAPE = \frac{100\%}{n} \sum_{i=1}^n \left| \frac{y_i^{\wedge} - y_i}{y_i} \right| \quad (14)$$

dimana:

$y_i^{\wedge}$  : Nilai prediksi

$y_i$  : Nilai aktual

$n$  : Jumlah data

### 3. Mean Square Error (MSE)

*Mean Square Error* adalah penjumlahan nilai kuadrat error atau selisih dari nilai aktual dan nilai prediksi, kemudian membagi jumlah nilai tersebut dengan banyaknya waktu dari sebuah data peramalan atau dapat dirumuskan pada persamaan sebagai berikut [33]:

$$MSE = \frac{\sum (\text{Aktual} - \text{Prediksi})^2}{n} \quad (15)$$

### 4. Mean Absolute Error (MAE)

*Mean absolute error* adalah nilai kesalahan rata-rata yang error dari nilai aktual dengan nilai prediksi. MAE secara umum digunakan untuk pengukuran prediksi error pada analisis *time series* [35]. MAE dapat dirumuskan pada persamaan sebagai berikut:

$$MAE = \sum \frac{|Y' - Y|}{n} \quad (16)$$

dimana:

$Y'$  : Nilai prediksi

$Y$  : Nilai Aktual

$n$  : Jumlah data

### 5. Mean Absolute Deviation (MAD)

*Mean absolute deviation* adalah nilai absolut pada penyimpangan data terhadap *mean*. MAD dapat dirumuskan pada persamaan sebagai berikut:

$$MAD = \frac{\sum |\text{Aktual} - \text{Prediksi}|}{n} \quad (17)$$

dimana:

Aktual : Nilai aktual  
 Prediksi : Nilai prediksi  
 n : Jumlah data

## 2.10 Literature Review

Dalam penyusunan Skripsi ini, terdapat beberapa penelitian terkait yang sebelumnya telah dilakukan oleh pihak lain. Adapun model dan data yang digunakan serta hasil-hasil dari penelitian akan dijadikan sebagai referensi dalam penyusunan Skripsi ini. Deskripsi singkat dari penelitian yang telah dilakukan dapat dilihat pada Tabel 2.1

Tabel 2.1 Studi Literatur

No	Referensi	Model	Kontribusi	Area Aplikasi	Hasil
1	[7]	Long Short-Term Memory	Prediksi Relatif humidity selama dua tahun (2008 – 2009)	Energi	Hasil penelitian ini didapat nilai akurasi klasifikasi 99,55% pada epoch 100.
2	[36]	Long Short-Term Memory	Prediksi tingkat malaria selama 12 bulan (Januari 2019 – 31 Desember 2019)	Kesehatan	Hasil Penelitian ini didapat nilai akurasi <i>Venkatapuram</i> sebesar 96,11%, <i>Aswaraopeta</i> sebesar 94,42%, <i>Khammam</i> sebesar 88,72% dan <i>Aswapuram</i> sebesar 93,26%.
3	[37]	Long Short-Term Memory	Prediksi tingkat covid-19 menggunakan CX Images	Kesehatan	Hasil penelitian ini didapat nilai akurasi 100% untuk data sebanyak 1061 CX images.
4	[9]	Long Short-Term Memory	Prediksi kenaikan bitcoin menggunakan data selama 2 tahun (22 mei 2016 – 19 mei 2018) dalam penelitian ini menghasilkan prediksi kenaikan harga bitcoin sebesar 1.23%	Ekonomi atau Keuangan	Hasil penelitian ini didapat nilai akurasi rata-rata sebesar 95,36% pada epoch 100

5	[38]	Long Short-Term Memory	Prediksi pencemaran udara menggunakan data selama lebih dari 6 tahun (1 Januari 2015 – 31 Oktober 2021)	Kesehatan	Hasil dari penelitian ini didapat nilai RMSE $PM_{10}$ (0.07237) pada epoch 100, RMSE $SO_2$ (0.05841) pada epoch 50, RMSE $CO$ (0.05474) pada epoch 100, RMSE $O_3$ (0.04465) pada epoch 50. RMSE $NO_2$ (0.0431) pada epoch 50.
---	------	------------------------	---	-----------	---



UNIVERSITAS  
MIKROSKIL