Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Perpustakaan Univeristas Multimedia Nusantara

BAB 2
LANDASAN TEORI

2.1 Vaksin Covid-19

Dengan adanya darurat kesehatan global yang ditetapkan oleh *World Health Organization* (WHO) setelah kemunculan suatu virus Covid-19 yang menyebarluas ke seluruh dunia, dilakukan program vaksinasi Covid-19 sebagai salah satu program untuk menghentikan pandemi di seluruh dunia [13]. Pada Januari 2021, pemerintah Indonesia melaksanakan program vaksinasi Covid-19 kepada masyarakat Indonesia secara gratis dengan tujuan dapat mengakhiri kondisi pandemi Covid-19. Pemberian vaksin Covid-19 ditujukan kepada semua masyarakat Indonesia dan dilakukan secara bertahap yang dimulai dari tenaga medis hingga anak-anak [14]. Hingga saat ini, program vaksinasi Covid-19 telah memasuki dosis 4 yang baru hanya diberikan kepada tenaga kesehatan terlebih dahulu [7]. Dilansir dari *website University Hospital Southampton* yang mempelajari tentang tujuh vaksin Covid-19 yang berbeda, pemberian vaksin yang selalu terbarukan bertujuan untuk menguatkan kembali imunitas yang ada didalam tubuh karena banyaknya varian baru virus Covid-19 yang ada sehingga membuat vaksin yang telah diterima menjadi kurang efektif [15].

2.2 Twitter

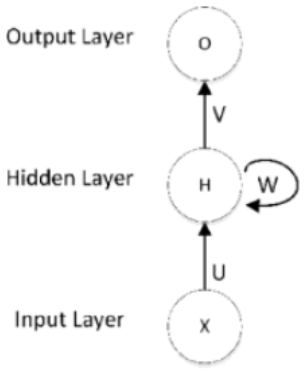
Twitter merupakan salah satu *platform micro-blogs* yang memiliki fitur dimana *user* dapat mengunggah sebuah komentar opini yang dinamakan dengan ”*tweet*”, dengan jumlah huruf mencapai 280 karakter per *tweet*. Selain membuat *tweet* yang menyampaikan sebuah komentar opini, seorang *user* juga dapat memberikan *like* dan *retweet* pada *tweet* milik *user* lain sebagai interaksi dengan sesama, menandai *tag* berbagai *user* lain dalam komentar, serta dapat memberikan respon kepada *tweet* milik *user* lain selama akun *user* tersebut bersifat publik yang dapat diakses oleh semua pengguna Twitter. Pada penelitian ini, data *tweet* akan dikumpulkan menggunakan Snscrape yang merupakan layanan untuk mengambil data dari berbagai media sosial, salah satunya Twitter.

2.3 Analisis Sentimen

Analisis Sentimen merujuk kepada suatu pemrosesan sebuah teks yang diekstraksi dan dikonversi untuk kemudian akan diklasifikasikan apakah kalimat tersebut memiliki sentimen yang positif, negatif, atau netral [16]. Penggunaan analisis sentimen sering digunakan untuk melihat respon konsumen terhadap suatu produk sehingga dapat diambil keputusan terkait bagaimana meningkatkan produk tersebut [17]. Analisis sentimen juga memiliki beberapa metode diantaranya analisis sentimen dengan metode *lexicon* yang melakukan analisa teks berdasarkan kamus dengan kata-kata yang sudah memiliki bobot sentimen, analisis sentimen dengan metode *machine learning*, dan analisis sentimen dengan metode *deep learning* [18].

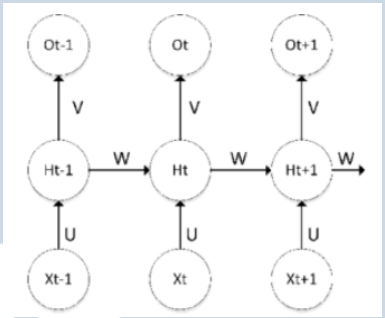
2.4 *Recurrent Neural Network*

Recurrent Neural Network (RNN) merupakan salah satu arsitektur jaringan syaraf tiruan yang memiliki kemampuan dalam mengingat informasi pola data pada arsitektur jaringannya sehingga mampu menangani pola-pola dari *input* data yang diberikan secara sekuensial [19]. Pada Gambar 2.1 merupakan arsitektur RNN, yang dimana *X* merupakan masukan kata per kata dan *O* merupakan hasil keluarannya. Pemrosesan dilakukan berulang kali secara sekuensial. Dalam setiap proses, keluaran kata akan disimpan oleh *H* dan akan dilanjutkan ke pemrosesan selanjutnya [20].



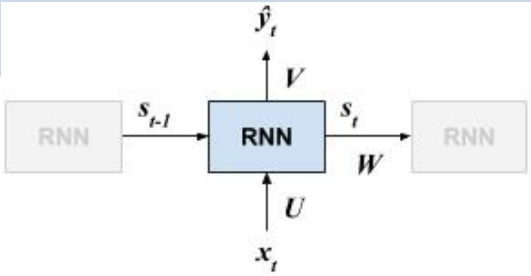
Gambar 2.1. Arsitektur RNN
Sumber: [20]

Gambar 2.2 merupakan penjabaran proses RNN dari masukan pertama hingga mendapat hasil keluaran terakhir. Penjabaran proses RNN ini memperlihatkan bahwa proses dilakukan secara berulang kali secara berantai.



Gambar 2.2. Penjabaran Arsitektur RNN

Sumber: [20]



Gambar 2.3. Alur Proses RNN

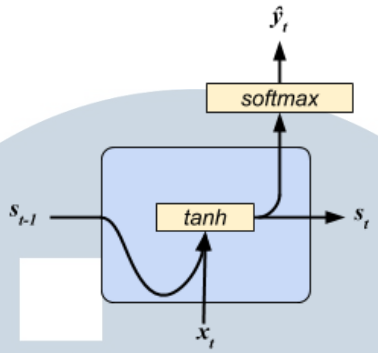
Sumber: [21]

Proses RNN yang terjadi pada H digambarkan seperti pada Gambar 2.3 yaitu dilakukan kalkulasi untuk menghasilkan *state* S_t berdasarkan *state* S_{t-1} dan masukan X_t, yang dimana akan dikalikan dengan parameter U dan W yang lalu selanjutnya akan dilanjutkan dengan pemrosesan menggunakan fungsi aktivasi *tanh*.

$$s_t = tanh(U \cdot x_t + W \cdot s_{t-1}) \quad (2.1)$$

$$\hat{\mathbf{y}}_t = softmax(V \cdot s_t) \quad (2.2)$$

Setelah dilakukan kalkulasi terhadap s_t, maka akan dilakukan kalkulasi untuk $\hat{y}_t$ sebagai keluaran dengan mengkalikan parameter V dengan s_t lalu diberikan fungsi aktivasi *softmax*. Berikut Gambar 2.4 sebagai bentuk visualisasi dan perhitungan yang dilakukan pada Rumus 2.2.



Gambar 2.4. *Output* Alur Proses RNN

Sumber: [21]

Dengan bertambah panjangnya suatu teks yang akan diproses, RNN memiliki keterbatasan yang menimbulkan suatu permasalahan yang disebut *vanishing gradient* yang dimana dengan seiring proses *back propagation* dilakukan, maka nilai dari bobot yang digunakan untuk melakukan pembaruan akan menyusut sehingga tidak akan meningkatkan hasil proses *learning*. Oleh sebab itu, digunakan suatu metode yang dapat membantu mengatasi permasalahan RNN yaitu dengan menggunakan metode *Long Short-Term Memory* (LSTM) [22].

2.5 *Long Short-Term Memory*

Long Short-Term Memory atau yang biasa disingkat dengan LSTM berfungsi untuk membantu mengatasi permasalahan *vanishing gradient* yang dimiliki oleh RNN. LSTM memiliki suatu memori yang disebut dengan *cell* yang diperoleh melalui *input state* sebelum dan *input state* saat ini. Dalam prosesnya, kumpulan *cells* akan menyimpan ataupun membuang informasi berdasarkan gerbang-gerbang (*gates*) yang dimiliki oleh LSTM [23]. Pada proses LSTM, terdapat beberapa *gates* yang disebut dengan *forget gate, input gate, output gate*, dan *memory cell* yang akan dikalkulasi menjadi nilai keluaran sebagai *hidden layer* yang akan diteruskan pada proses selanjutnya. Berikut merupakan beberapa rumus LSTM [24].

UNIVERSITAS MULTIMEDIA NUSANTARA

- *Forget Gate*

  Pada *forget gate*, nilai masukan (*input*) dan nilai keluaran (*output*) sebelunya akan digabungkan, dan fungsi aktivasi sigmoid yang akan menentukan apakah informasi akan dilupakan atau diingat [24].

$$f_t = \sigma(W_f \cdot [x_t + h_{t-1}] + b_f) \tag{2.3}$$

  - σ: *Sigmoid activation function*
  - f_t: *forget gate*
  - W_f: *weight forget gate*
  - x_t: *input cell*
  - h_{t-1}: *output cell sebelumnya*
  - b_f: *bias forget gate*

- *Input Gate*

  Pada *Input gate*, terdapat dua bagian dimana proses melalui fungsi aktivasi sigmoid yang menerima nilai masukan (*input*) dan nilai keluaran (*output*) dari sebelunya akan digabungkan dan akan mendapatkan nilai *input*. Selanjutnya, jalur dengan fungsi aktivasi *tanh* dilakukan untuk mendapatkan nilai *candidate memory cell* [24].

$$i_t = \sigma(W_i \cdot [x_t + h_{t-1}] + b_i) \tag{2.4}$$

$$\tilde{C}_t = tanh(W_C \cdot [x_t + h_{t-1}] + b_C) \tag{2.5}$$

  - i_t: *input gate*
  - *tanh*: *Tanh activation function*
  - W_i: *weight input gate*
  - b_i: *bias input gate*
  - $\tilde{C}_t$: *candidate memory cell*
  - W_C: *weight candidate*
  - b_C: *bias candidate*

- *Cell State*

  Cell state C_t didapat dari penggabungan antara nilai perkalian dari *forget gate*

dengan nilai *state* sebelumnya f_t dengan nilai *candidate memory cell* yang dikalikan dengan nilai *input gate* [24].

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \tag{2.6}$$

- C_t: *cell state*
- C_{t-1}: *candidate state sebelumnya*

- *Output Gate*

  Output gate akan menghasilkan nilai keluaran yang diproses dari hasil pembaharuan *cell* serta diberikan fungsi aktivasi *sigmoid* [24].

$$O_t = \sigma(W_o \cdot [x_t + h_{t-1}] + b_o) \tag{2.7}$$

  - O_t: *output gate*
  - W_o: *weight output gate*
  - b_o: *bias output gate*

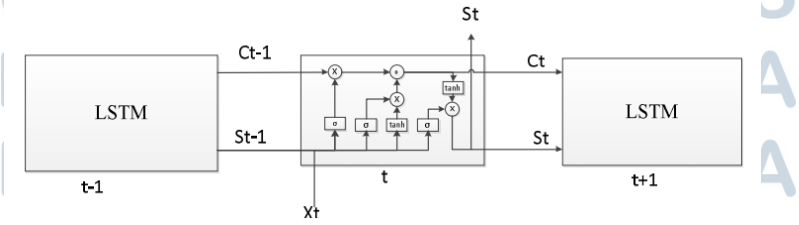
- *Hidden State*

  Pada *hidden state*, nilai diambil dari *output gate* lalu dikalikan dengan nilai *cell state* yang diberikan fungsi aktivasi *tanh*. Nilai pada *hidden state* akan berfungsi untuk dibawa ke proses selanjutnya [24].

$$h_t = O_t \cdot tanh(C_t) \tag{2.8}$$

  - h_t: *hidden state*

Gambaran terkait bentuk arsitektur pada metode LSTM dapat dilihat pada Gambar 2.5.



Gambar 2.5. Arsitektur *Long Short-Term Memory*

Sumber: [24]

2.6 *Text Pre-Processing*

Text Pre-Processing merupakan salah satu tahapan awal yang umum digunakan untuk mengolah dataset bertipe teks sehingga tercipta suatu data yang berkualitas dan siap digunakan untuk proses selanjutnya [25]. Sebelum menjadi data yang berkualitas, terdapat beberapa tahapan dalam melakukan *text pre-processing* antara lain yaitu:

1. *Case Folding*, yaitu tahapan yang dilakukan untuk mengubah huruf alfabet dari yang semula huruf kapital menjadi huruf kecil.
2. *Stopword Removal*, yaitu tahapan yang dilakukan untuk menghilangkan kata yang tidak memiliki makna dalam proses klasifikasi.
3. *Symbol Removal*, yaitu tahapan untuk menghapus karakter khusus seperti tanda baca pada teks kecuali spasi.
4. *Tokenization*, yaitu tahapan untuk memecah suatu komentar berbentuk kalimat menjadi satuan kata.
5. *Stemming*, yaitu proses mengubah suatu teks yang memiliki imbuhan menjadi kata dasar.

2.7 *Confusion Matrix*

Salah satu cara untuk mengukur suatu kinerja dari sebuah sistem klasifikasi yaitu dengan menggunakan *confusion matrix*. *Confusion matrix* merupakan sebuah tabel yang memiliki 4 kombinasi hasil sebagai pengukuran kerja yaitu *True Positive*, *True Negative*, *False Positive*, dan *False Negative* [24].

Penulisan tabel *confusion matrix* dapat dilihat pada Tabel 2.1

Tabel 2.1. Tabel *Confusion Matrix*

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Sumber: [26]

Berikut merupakan penjelasan dari Tabel 2.1 [26]:

1. *True Positive* merupakan hasil prediksi positif dan dan hal tersebut terdeteksi dengan benar.
2. *True Negative* merupakan hasil prediksi bernilai negatif yang terdeteksi dengan benar.
3. *False Positive* adalah hasil prediksi yang positif namun hal tersebut salah.
4. False Negative adalah hasil prediksi negatif dan hal itu salah.

Setelah mendapatkan hasil dari *confusion matrix*, maka dapat dilakukan penghitungan *Recall*, *Precision*, *F-1 Score*, dan *Accuracy*.

Perhitungan *recall* akan memberikan persentase keberhasilan model untuk menemukan kembali informasi [26].

$$Recall = \frac{TP}{(TP+FN)} \tag{2.9}$$

Perhitungan *accuracy* memberikan persentase hasil seberapa akurat model memprediksi dengan benar [26].

$$Accuracy = \frac{(TP+TN)}{(TP+FN+TN+FP)} \tag{2.10}$$

Perhitungan *precision* menghasilkan persentase hasil antara data yang diminta dengan hasil prediksi dari model [26].

$$Precision = \frac{TP}{(TP+FP)} \tag{2.11}$$

Perhitungan *F-1 Score* akan memberikan hasil rata-rata dari nilai *recall* dan nilai *precision* [26].

$$F\text{-}1\ Score = \frac{2 \times precision \times recall}{precision + recall} \tag{2.12}$$

UNIVERSITAS MULTIMEDIA NUSANTARA