# ANALISIS SENTIMEN PADA STARTUP EDUTECH MENGGUNAKAN CNN-BiLSTM

## Kelompok 5


Sudhanta Dwitama (1806208314)
Syamsyuriani (1806207942)
Lenni Fitri Anwar (1806144462)
Lista Kurniawati (1806208011)

# OUTLINE

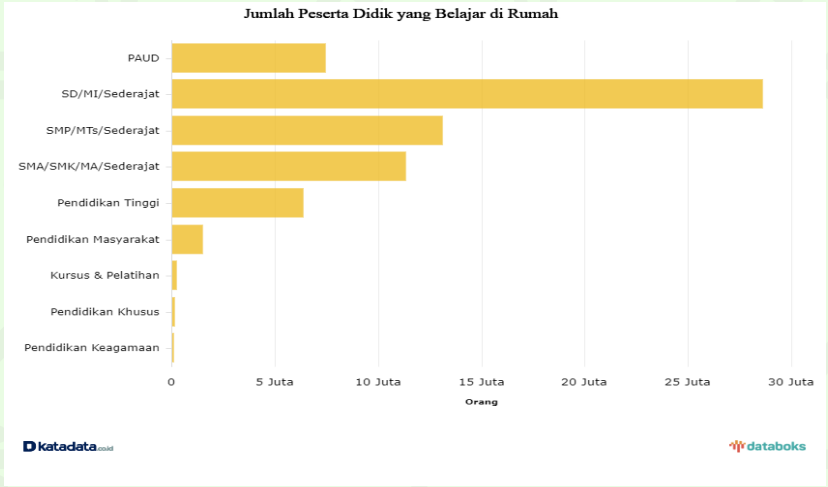
01 Pendahuluan

02 Metode CNN-BiLSTM

03 Simulasi & Analisis Hasil

04 Kesimpulan

# Pendahuluan




Pada tahun 2020, berkenaan dengan penyebaran Coronavirus Disease (COVID-19) yang semakin meningkat, Kemendikbud RI menetapkan proses belajar dari rumah melalui daring atau jarak jauh [1].

Terjadi perubahan cara belajar secara signifikan dari pembelajaran luring menjadi pembelajaran daring (PJJ). Hal ini meningkatkan pertumbuhan startup edutech di Indonesia.

Selama pandemi virus corona, jumlah pengguna platform pendidikan melonjak. Ruangguru misalnya, mencatatkan peningkatan jumlah konsumen 50% menjadi 22 juta pada tahun lalu [2].

Quipper co·learn zenius EDUCATION ruang guru

Tercatat pada Juni 2020, sudah terdapat 44 pemain edutech di Indonesia dan diperkirakan akan terus bertambah. Beberapa yang cukup populer di masyarakat, diantaranya Quipper, Zenius, Ruangguru, IndonesiaX, Cakap, dan masih banyak lagi [3].

Melihat kepopuleran startup edutech ini, membuat banyak pihak mulai melirik data pada platform pendidikan untuk diolah menjadi informasi berharga yang dapat digunakan untuk berbagai kepentingan penelitian. Salah satu penelitian yang memanfaatkan data dari edutech adalah sentimen analisis [4].

Algoritma machine learning merupakan suatu algoritma yang belajar dari data [10]. Salah satu kelas dalam machine learning adalah deep learning [11]. Deep learning menggunakan deep neural network untuk menyelesaikan masalah pada domain machine learning.

Model klasifikasi sederhana dapat digunakan untuk melakukan analisis sentimen deep learning. Dua arsitektur Deep Learning yang sering digunakan untuk analisis sentimen adalah Convolutional Neural Network (CNN) dan Long Short-Term Memory (LSTM). Dalam penelitian ini digunakan model CNN-BiLSTM [6].

# Definisi Masalah

Masalah yang diangkat pada penelitian kali ini adalah analisis sentimen pada start up edutech dengan menggunakan metode CNN-BiLSTM.

Dalam sudut pandang machine learning, masalah analisis sentimen dengan metode CNN-BiLSTM ini termasuk dalam masalah klasifikasi. Tipe klasifikasi yang digunakan adalah klasifikasi biner, yaitu memiliki sentimen positif atau sentimen negatif.

# Related Works

(Chiu et. al., 2016) menemukan ide untuk membuat suatu model CNN-BiLSTM untuk NER (Named Entity Recognition).

(Shen et. al., 2017) menggabungkan model CNN BiLSTM dalam menyelesaikan permasalahan analisis sentimen untuk review film, dengan tingkat akurasi 89,7% yang lebih baik dibandingkan hanya menggunakan CNN saja atau LSTM saja.

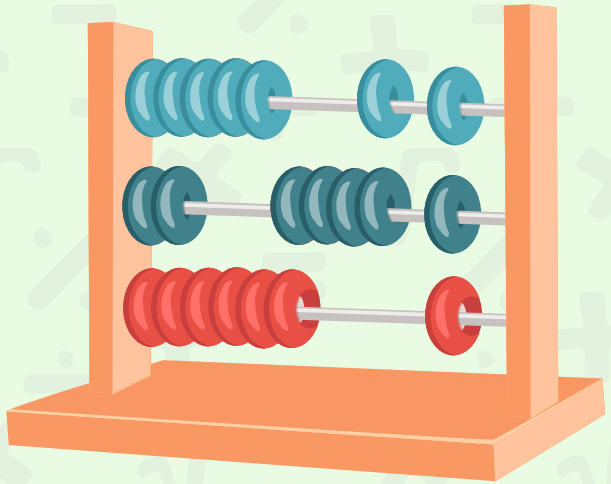
(Wijaya, 2020) membandingkan model LSTM-CNN dan BiLSTM-CNN untuk melakukan analisis sentimen berbahasa Indonesia. Didapatkan hasil rata-rata akurasi berturut-turut 87,881% dan 87,855%.

(Yue, Wang dan Lie Li, 2020) membandingkan model LSTM, CNN, BiLSTM, CNN-LSTM, dan CNN-BiLSTM untuk melakukan analisis sentimen. Dengan tingkat akurasi model berturut-turut 79,83%, 81,25%, 85,69%, 87,44%, 91,48%.

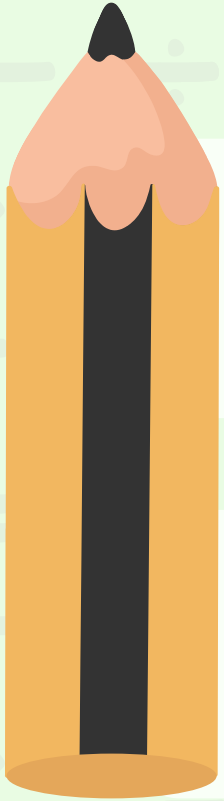
# Rumusan Masalah

1. Bagaimana mengklasifikasikan sentimen dari review aplikasi edutech di Indonesia menggunakan CNN-BiLSTM?

2. Bagaimana performa model CNN-BiLSTM untuk analisis sentimen startup edutech di Indonesia?



# Tujuan Penelitian



Mengklasifikasikan sentimen dari review aplikasi edutech di Indonesia menggunakan CNN-BiLSTM

Menganalisis performa model CNN-BiLSTM untuk analisis sentimen startup edutech di Indonesia

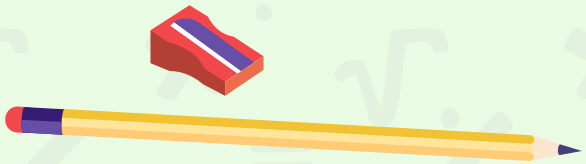
# Batasan Masalah

1. Data yang digunakan adalah dataset berupa reviews dari aplikasi Quipper di Google Play Store.
2. Model yang akan digunakan dalam penelitian ini adalah CNN-BiLSTM.
3. Data yang digunakan berbahasa Indonesia yang akan diberikan label berupa sentimen positif dan sentimen negatif.
4. Performa model CNN-BiLSTM diukur berdasarkan tingkat akurasi model.

# Metode CNN-BiLSTM

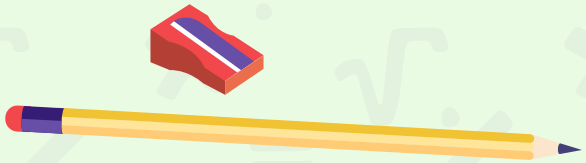
# CNN

*Convolution Neural Networks* (CNN) merupakan suatu *feedforward neural network* dengan lapisan *convolution* dan *pooling* yang digunakan untuk mempelajari representasi data dengan fitur berupa area dua dimensi [6].



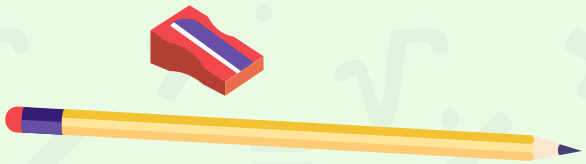
# LSTM

*Long Short-Term Memory* (LSTM atau BiLSTM) *Neural Networks* merupakan *neural network* dengan beberapa lapisan LSTM yang digunakan untuk mempelajari representasi data dengan memperhatikan urutan fitur dari data tersebut [6].

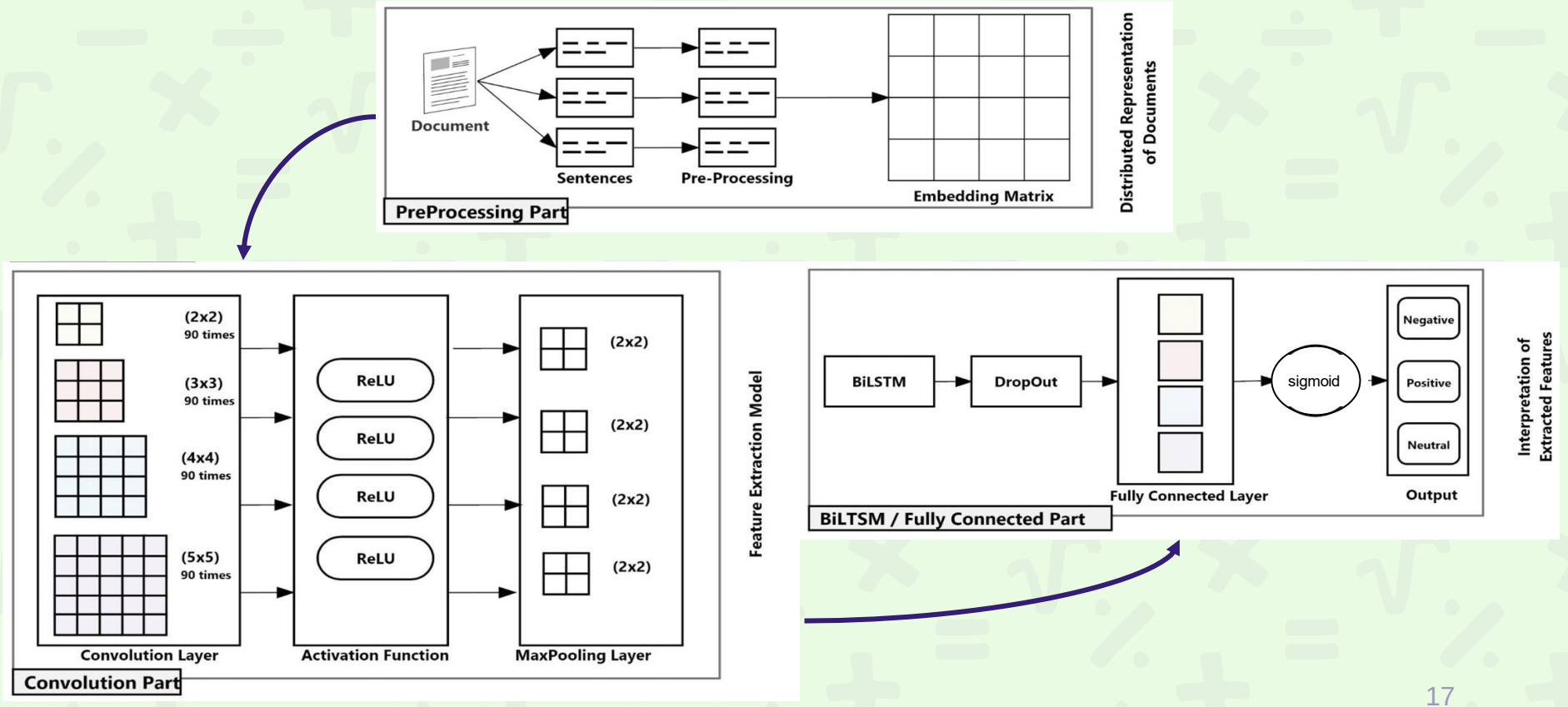


# CNN-BiLSTM

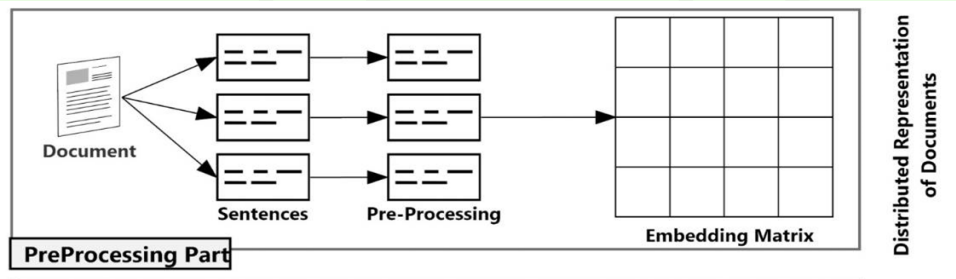
CNN-BiLSTM merupakan gabungan dua neural networks, yaitu CNN dan BiLSTM. Pengkombinasian ini dilakukan untuk menguji adaptasi CNN dengan BiLSTM, karena BiLSTM dikenal karena kinerjanya dalam analisis opini.



# Metode


Document
Sentences
Pre-Processing
Embedding Matrix
PreProcessing Part
Distributed Representation of Documents
(2x2) 90 times
(3x3) 90 times
(4x4) 90 times
(5x5) 90 times
Convolution Layer
ReLU
ReLU
ReLU
ReLU
Activation Function
(2x2)
(2x2)
(2x2)
(2x2)
MaxPooling Layer
Convolution Part
Feature Extraction Model
BiLSTM
DropOut
Fully Connected Layer
sigmoid
Negative
Positive
Neutral
Output
BiLTSM / Fully Connected Part
Interpretation of Extracted Features

# Metode



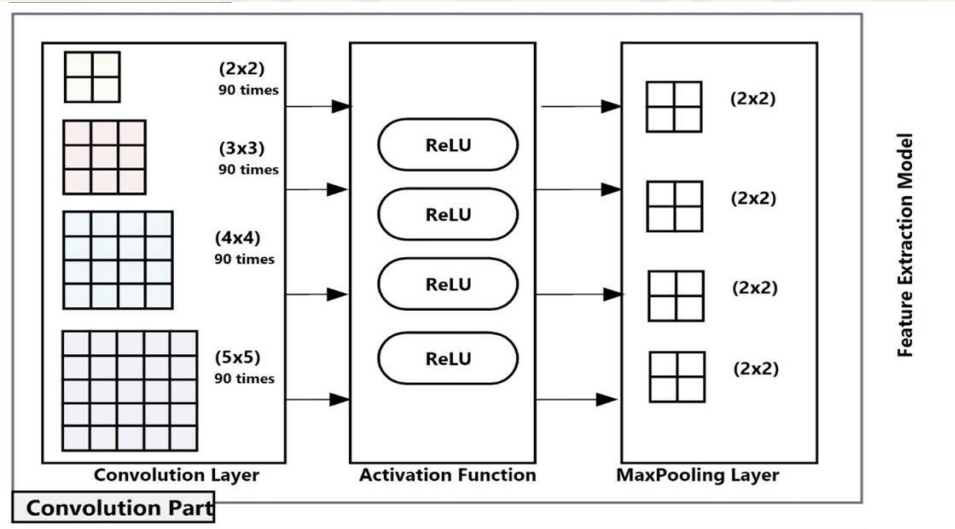

## Input

*Input* yang diterima model berupa kalimat dengan sepanjang $n$ kata.

## Word Embedding

*Word Embedding* digunakan untuk memetakan setiap kata pada input menjadi vektor berdimensi $d$. Hasilnya, didapatkan n vektor $x_i$ berdimensi $d$ yang masing-masing mempresentasikan kata ke-$i$ dari kalimat.

# Metode



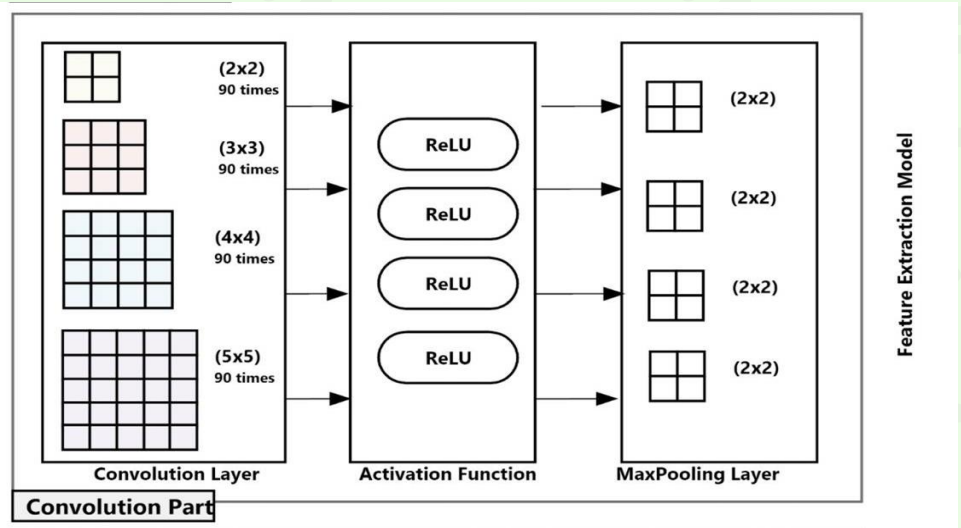

## Convolution layer

Convolution layer adalah layer pertama yang dilalui oleh input dan bertugas mencari fitur-fitur penting di tiap penggalan kalimat dengan panjang tertentu. Pada convolution layer, input yang sudah berbentuk matriks akan diproses dengan region size dan jumlah filter yang ditentukan. Misalkan region size $s$ dan jumlah filter $l$, maka dihasilkan output $l$ vektor, $m_1, \ldots, m_l$, berdimensi $n-s+1$. Untuk bagian CNN, fungsi aktivasi yang digunakan adalah fungsi ReLu.

# Metode

## *MaxPooling Layer*

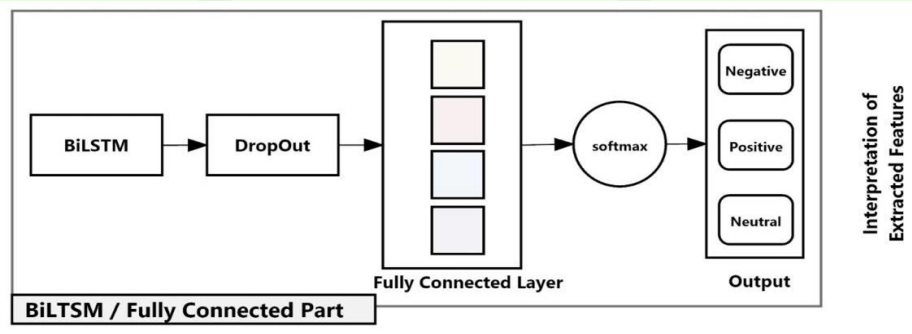
*Pooling Layer* digunakan untuk mengambil fitur paling penting dari semua fitur yang diperoleh. *MaxPooling* dilakukan dengan mengambil nilai maksimum dari semua nilai fitur yang diperoleh. Pada bagian ini akan diambil fitur terpenting dari masing-masing vektor $m_1, \dots, m_l$ , yaitu elemen dengan nilai terbesar dari tiap vektor. Misalkan dari vektor *mi* diambil fitur terpenting *p*, dengan $p = \max(m_i)$. Dari tiap vektor $m_1, \dots, m_l$ , dihasilkan nilai skalar $p_1, \dots, p_l$ yang apabila digabung akan membentuk vektor tunggal $p$ berdimensi $l$.




## *Output maxpooling layer / Input* BiLSTM

Selanjutnya, output berupa vektor tunggal $p$ dari layer sebelumnya akan diproses oleh layer LSTM. Cell LSTM akan memproses vektor tunggal $p$.

# Metode




## BiLSTM *Layer*

Bi-LSTM merupakan pengembangan dari LSTM dengan menggabungkan dua layer LSTM, namun keduanya menerima input dengan arah yang berbeda. Satu layer LSTM menerima input dimulai dari kata pertama, sedangkan layer LSTM yang lain menerima input dari akhir kata. Dimensi dari output yang dihasilkan layer BiLSTM ditentukan berdasarkan banyaknya unit BiLSTM ($k$). input berupa vektor tunggal $p$ berdimensi $l$ akan menghasilkan output vektor tunggal $h$ berdimensi $k$.

## *Output* Bi-LSTM

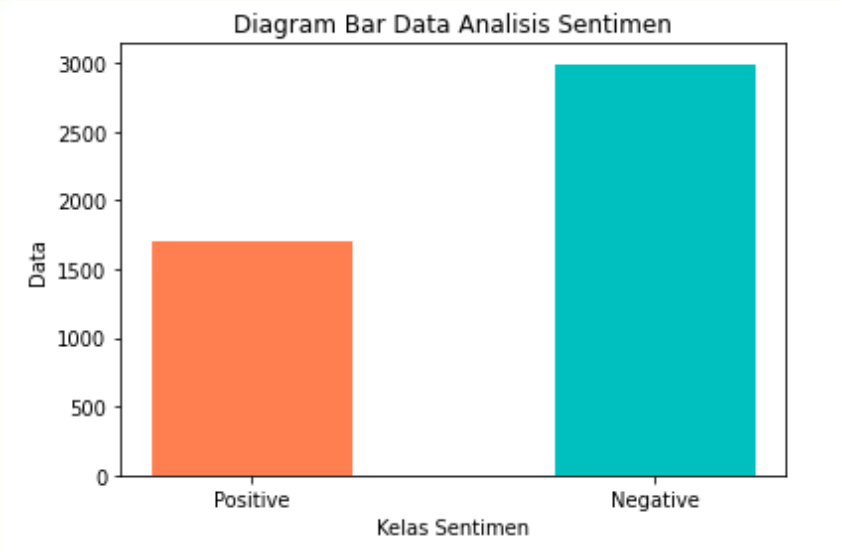
Tahap selanjutnya, vektor tunggal $h$ akan diproses oleh *fully connected layer*. Pada layer ini vektor $h$ dengan dimensi $k$ akan dipetakan ke dua output nodes dengan menggunakan fungsi aktivasi sigmoid.

# Simulasi & Analisis Hasil

# Dataset

| | review | label |
|---|---|---|
| 0 | Apk Quipper sangat membantu dalam memahami pel... | -1 |
| 1 | Penjelasannya bagus, tutornya juga enak ngejel... | -1 |
| 2 | Tolong dong quipper, saya lagi ngerjain soal a... | -1 |
| 3 | Banyak bug. klo matematika, fisika, kimia ,Jaw... | -1 |
| 4 | Aplikasinya kurang menarik, pembahasan nya lam... | -1 |
| ... | ... | ... |
| 4682 | Gimana cara masukin soal? | -1 |
| 4683 | Amit jangan donwlod nyesel | -1 |
| 4684 | cari kode kelasnya dimanaaa.. | -1 |
| 4685 | Harus bayar paketnya | -1 |
| 4686 | Cara nya bagaimana ya? | -1 |

4687 rows × 2 columns




Dataset diambil dari Google Play Store yang berjumlah 4687 sentimen.
Dataset terdiri dari 1696 sentimen positif dan 2991 sentimen negatif.

# Pembersihan Data

```python
def clean_text(tweet):
    # Convert to lower case
    tweet = tweet.lower()
    # Clean www.* or https?://*
    tweet = re.sub('((www\.[^\s]+)|(https?://[^\s]+))','',tweet)
    # Clean @username
    tweet = re.sub('@[^\s]+','',tweet)
    #Remove punctuation
    tweet = re.sub(r'[^\w\s]',' ', tweet)
    #Replace #word with word
    tweet = re.sub(r'#([^\s]+)', r'\1', tweet)
    #Clean number
    tweet = re.sub(r'[\d-]', '', tweet)
    #Remove additional white spaces
    tweet = re.sub('[\s]+', ' ', tweet)
    #trim
    tweet = tweet.strip('\'"')
    # Clean per Words
    words = tweet.split()
    tokens=[]
    for ww in words:
        #split repeated word
        for w in re.split(r'[-/\s]\s*', ww):
            #replace two or more with two occurrences
            pattern = re.compile(r"(.)\1{1,}", re.DOTALL)
            w = pattern.sub(r"\1\1", w)
            #strip punctuation
            w = w.strip('\'"?,.')
            #check if the word cosists of two or more alphabets
            val = re.search(r"^[a-zA-Z][a-zA-Z][a-zA-Z]*$", w)
            #add tokens
            if(w in stopwords or val is None):
                continue
            else:
                tokens.append(w.lower())

    tweet = " ".join(tokens)
    return tweet
```

Dilakukan pembersihan data dengan langkah-langkah berikut:

1. Mengubah kalimat menjadi huruf kecil
2. Menghilangkan tautan web
3. Menghilangkan *username*
4. Menghilangkan tanda baca
5. Menghilangkan tagar
6. Menghilangkan angka
7. Memisahkan kata yang disambung oleh tanda baca '-'
8. Menghapus huruf berulang lebih dari dua yang bersebelahan menjadi hanya dua
9. dll.

# Pembersihan Data

| | |
|---|---|
| Sebelum | Apk Quipper sangat membantu dalam memahami pelajaran dengan baik apalagi ada Solusi quippernya , tpi terkadang didalam pembahasan ada yg tidak dimengerti:(, seandainya ada room tanya di pembahasan supaya bisa full mengerti. Terimakasih Quipper |
| Sesudah | apk quipper membantu memahami pelajaran solusi quippernya tpi terkadang didalam pembahasan yg dimengerti seandainya room pembahasan full mengerti terimakasih quipper |

# Tokenisasi dan Sequencing

Tokenisasi adalah proses di mana setiap kata unik pada data akan diurutkan berdasarkan kata yang paling sering muncul. Kemudian urutan tersebut digunakan sebagai indeks kata.

Proses sequencing, yaitu proses di mana kalimat yang terdiri kata-kata tersebut diubah menjadi barisan indeks kata.

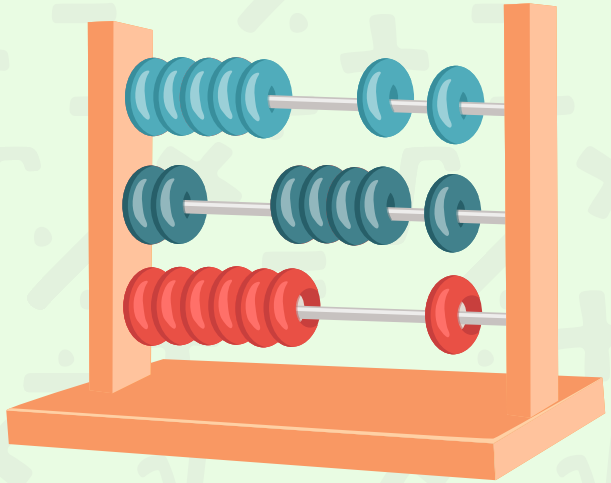
Pada program ditambahkan pad sequence = 80 artinya batas kata maksimal di suatu kalimat adalah 80.

```
array([   0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,    0,    0,    0,    0,    0,    0,
          0,    0,    0,    0,    0,   30,    1,   13,  177,   43,  322,
        354,   73,  461, 2366,  100,    8,  279, 1602, 1222,  100,  414,
        162,   63,    1], dtype=int32)
```

# Pembagian Dataset

```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.2, random_state = 42)
```

Dataset dipisahkan dengan rasio *training:testing* = 80:20.



# Optimasi Parameter

Optimasi parameter dilakukan dengan menggunakan metode Bayesian Optimization. Metode Bayesian Optimization akan mencoba beberapa kombinasi kemungkinan kandidat parameter yang digunakan.

Metode bayesian optimization adalah metode optimasi global untuk fungsi *black-box*. Bayesian optimization menggunakan *acquisition function* untuk menyeimbangkan antara eksplorasi dan eksploitasi pada penentuan nilai hyperparameter berikutnya. Algoritma ini memiliki performa yang lebih baik dari pendekatan optimasi yang lain [13].

| Parameter-parameter pada model | |
|---|---|
| **Embedding Size** | Ukuran dimensi vektor kata. Embedding size merupakan salah satu parameter yang akan dioptimasi. |
| **Number of filter CNN** | Jumlah filter yang digunakan pada convolutional layer. Number of filter CNN merupakan salah satu parameter yang akan dioptimasi. |
| **Number of unit LSTM** | Dimensi vektor output pada layer LSTM. Number of unit LSTM merupakan salah satu parameter yang akan dioptimasi. |
| **Region Size** | Ukuran region pada convolutional layer. Region size merupakan salah satu parameter yang akan dioptimasi. |
| **Activation** | Fungsi aktivasi yang digunakan pada CNN layer adalah fungsi ReLU, sedangkan fungsi aktivasi yang digunakan pada fully connected layer adalah fungsi sigmoid. |
| **Learning Rate** | Digunakan untuk mempercepat laju pembelajaran dalam menentukan bobot yang optimal. |
| **Regularisasi L2** | Teknik yang sering digunakan untuk regularisasi model. Parameter regularisasi L2 ini ada pada CNN layer, LSTM layer, dan fully connected layer. Parameter ini akan dioptimasi untuk masing-masing layer. |
| **Epoch** | Jumlah pengulangan proses pembelajaran dalam satu data set. Epoch yang akan digunakan dalam penelitian ini adalah 10 [12]. |
| **Batch Size** | Jumlah sampel data yang diproses pada satu epoch. Batch size yang akan digunakan dalam penelitian ini adalah 32 [6]. |

Akan digunakan metode bayesian optimization dengan maksimum trial 10 kombinasi untuk mencari kombinasi hyperparameter terbaik model CNN-BiLSTM.

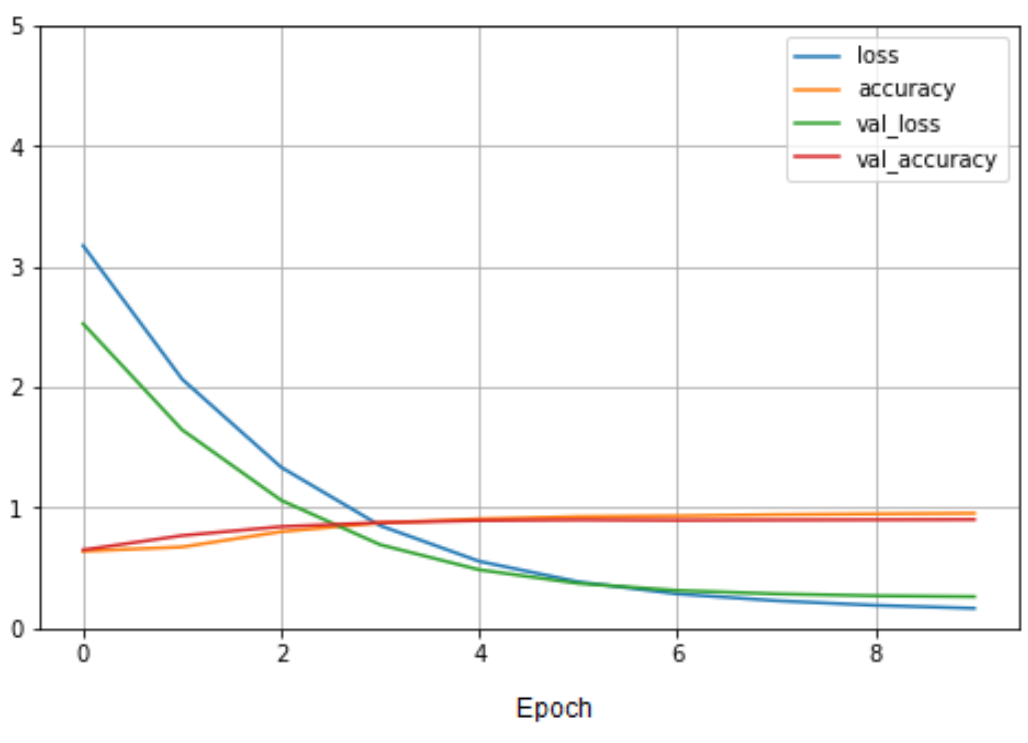
| Parameter yang akan dioptimasi pada model CNN-BiLSTM | | | |
|---|---|---|---|
| **Layer** | **Parameter** | **Nilai** | **Rujukan** |
| Word Embedding | Embedding Size | 64; 100; 128 | [6] |
| CNN | Number of filter | 200; 250; 300 | |
| | Region Size | 3; 4; 5 | |
| | L2 CNN | 0; 0,01 | |
| LSTM | Number of unit | 100; 150; 200 | |
| | L2 Kernel | 0; 0,01 | |
| | L2 Recurrent | 0; 0,01 | |
| Fully Connected | L2 Dense | 0; 0,01 | |
| Optimasi Adam | Learning rate | 0,01; 0,001; 0,0001 | |

# Hasil parameter yang telah dioptimasi pada model CNN-BiLSTM

Setelah dilakukan sebanyak 5 kali percobaan diperoleh kombinasi *hyperparameter* dengan akurasi terbaik sepeti pada tabel di samping.

| Parameter | Hasil |
|---|---|
| *Embedding Size* | 128 |
| *Number of filter* | [250, 250, 200, 250] |
| *Region Size* | [4, 4, 3, 3] |
| *L2 CNN* | [0, 0.01, 0.01, 0.01] |
| *Number of unit* | [200, 200] |
| *L2 Kernel* | [0, 0.01] |
| *L2 Recurrent* | [0.01, 0.0] |
| *L2 Dense* | 0 |
| *Epoch* | 10 |
| *Learning Rate* | 0.0001 |
| **Loss** | **0.2624** |
| **Akurasi** | **0.9023** |

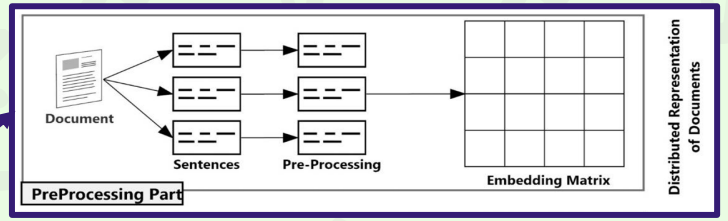
# Grafik *loss* dan *accuracy*

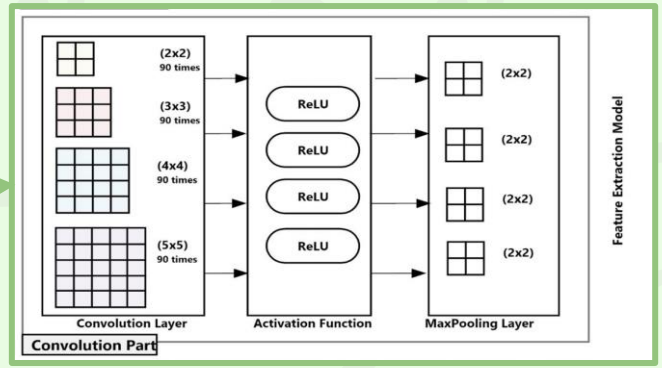


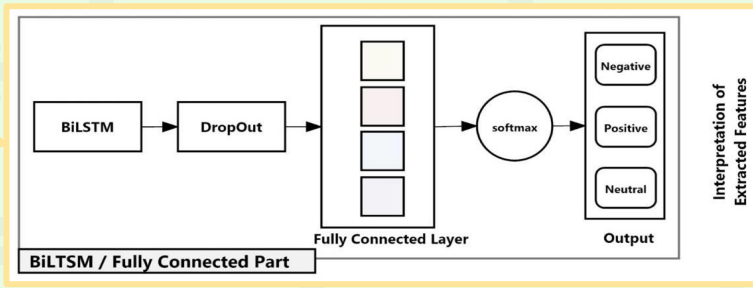

Terlihat bahwa nilai *loss* data *training* **menurun** dan nilai *loss* data validasi juga **cenderung menurun**. Sehingga model yang diperoleh tidak *overfit*.

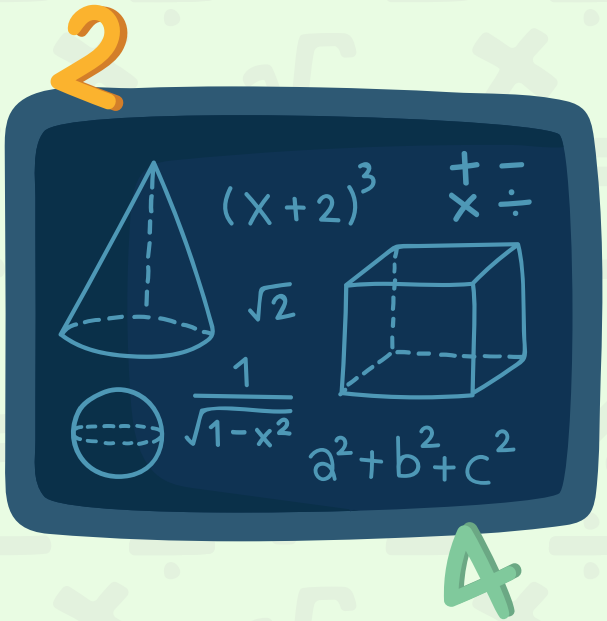
# Model CNN-BiLSTM

```
Model: "model"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   Connected to
=================================================================
input_5 (InputLayer)         [(None, 80)]              0
_________________________________________________________________
embedding_4 (Embedding)      (None, 80, 100)           568000    input_5[0][0]
_________________________________________________________________
conv1d (Conv1D)              (None, 76, 200)           100200    embedding_4[0][0]
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 76, 200)           100200    embedding_4[0][0]
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 76, 200)           100200    embedding_4[0][0]
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 76, 200)           100200    embedding_4[0][0]
_________________________________________________________________
global_max_pooling1d (GlobalMax (None, 200)            0         conv1d[0][0]
_________________________________________________________________
global_max_pooling1d_1 (GlobalM (None, 200)            0         conv1d_1[0][0]
_________________________________________________________________
global_max_pooling1d_2 (GlobalM (None, 200)            0         conv1d_2[0][0]
_________________________________________________________________
global_max_pooling1d_3 (GlobalM (None, 200)            0         conv1d_3[0][0]
_________________________________________________________________
concatenate (Concatenate)    (None, 800)               0         global_max_pooling1d[0][0]
                                                                 global_max_pooling1d_1[0][0]
                                                                 global_max_pooling1d_2[0][0]
                                                                 global_max_pooling1d_3[0][0]
_________________________________________________________________
layer (Layer)                (None, 800)               0         concatenate[0][0]
_________________________________________________________________
layer_1 (Layer)              (None, 800)               0         layer[0][0]
_________________________________________________________________
dropout (Dropout)            (None, 800)               0         layer_1[0][0]
_________________________________________________________________
dense (Dense)                (None, 1)                 801       dropout[0][0]
=================================================================
Total params: 969,601
Trainable params: 969,601
Non-trainable params: 0
```



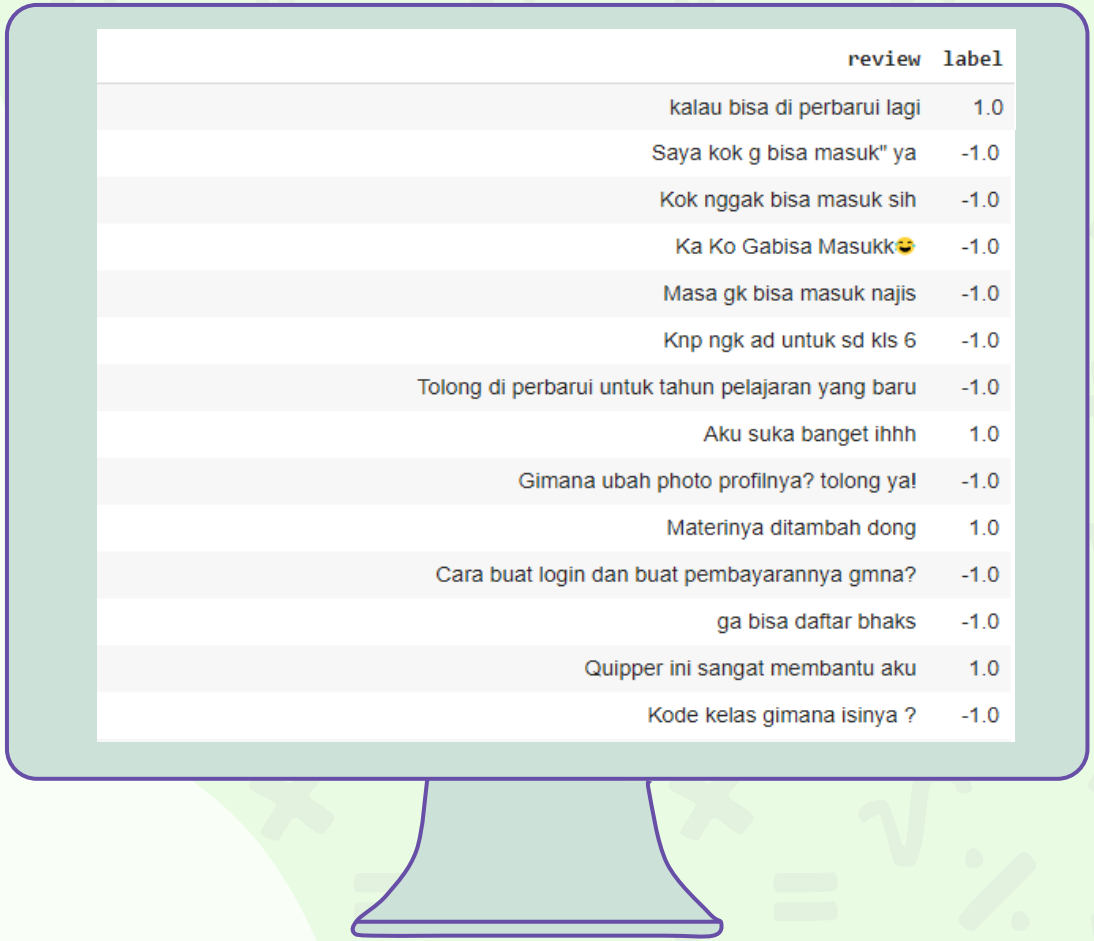



# Analisis Hasil

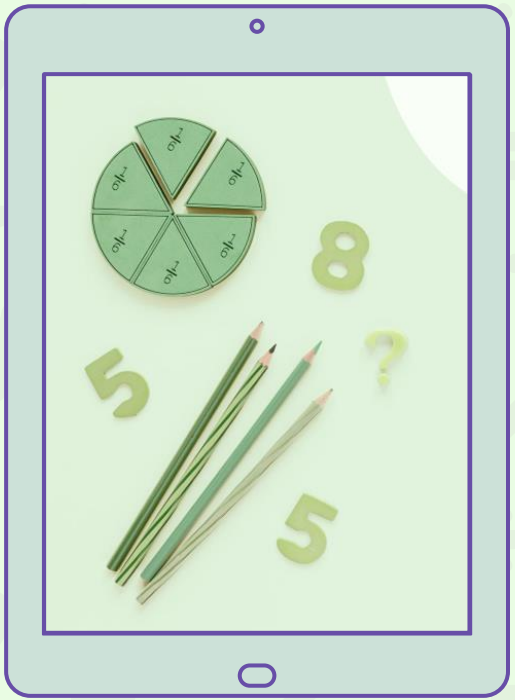
Performa akurasi hasil yang diperoleh dari model yang digunakan sebesar 90,22556%

# Prediksi Data Baru

Gambar disamping adalah potongan hasil prediksi model pada data baru yang belum di beri label. Terlihat bahwa model cukup baik dalam memprediksi label dari data baru yang dimasukkan



| review | label |
|---|---|
| kalau bisa di perbarui lagi | 1.0 |
| Saya kok g bisa masuk" ya | -1.0 |
| Kok nggak bisa masuk sih | -1.0 |
| Ka Ko Gabisa Masukk😎 | -1.0 |
| Masa gk bisa masuk najis | -1.0 |
| Knp ngk ad untuk sd kls 6 | -1.0 |
| Tolong di perbarui untuk tahun pelajaran yang baru | -1.0 |
| Aku suka banget ihhh | 1.0 |
| Gimana ubah photo profilnya? tolong ya! | -1.0 |
| Materinya ditambah dong | 1.0 |
| Cara buat login dan buat pembayarannya gmna? | -1.0 |
| ga bisa daftar bhaks | -1.0 |
| Quipper ini sangat membantu aku | 1.0 |
| Kode kelas gimana isinya ? | -1.0 |



# Kesimpulan

Pada penelitian ini telah dilakukan analisis sentimen pada dataset berupa kumpulan review berbahasa indonesia dari aplikasi Quipper di Google Play Store. Berdasarkan hasil simulasi yang telah dilakukan disimpulkan bahwa kinerja model CNN-BiLSTM menghasilkan akurasi sebesar 90,23%.

# Saran

Penelitian lebih lanjut yang dapat dilakukan mengenai masalah analisis sentimen pada Start Up Edutech di Indonesia adalah:

1. Menggunakan model klasifikasi lain seperti SVM, Naive-Bayes, dll.
2. Menambahkan jumlah dataset yang digunakan.

# Daftar Pustaka

[1] SURAT EDARAN MENDIKBUD NO 4 TAHUN 2020 TENTANG PELAKSANAAN KEBIJAKAN PENDIDIKAN DALAM MASA DARURAT PENYEBARAN CORONA VIRUS DISEASE (COVID- 1 9) – Pusdiklat Pegawai Kementerian Pendidikan dan Kebudayaan. (2020). Retrieved April 27, 2021, from https://pusdiklat.kemdikbud.go.id/surat-edaran-mendikbud-no-4-tahun-2020-tentang-pelaksanaan-kebijakan-pendidikan-dalam-masa-darurat-penyebaran-corona-virus-disease-covid-1-9/

[2] Pandemi Masih Berlanjut, Startup Pendidikan Sasar Pengguna di Desa - Startup Katadata.co.id. (2021). Retrieved April 27, 2021, from https://katadata.co.id/desysetyowati/digital/601bad025a0c2/pandemi-masih-berlanjut-startup-pendidikan-sasar-pengguna-di-desa

[3] Mengetahui Prospek & Perkembangan Bisnis Edutech di Masa PSBB - IDS Digital College. (2020). Retrieved April 27, 2021, from https://ids.ac.id/mengetahui-prospek-perkembangan-bisnis-edutech-di-masa-psbb/

[4] Lin, Y., Wang, X., & Zhou, A. (2016). Opinion spam detection. Opinion Analysis for Online Reviews, May, 79–94. https://doi.org/10.1142/9789813100459_0007

# Daftar Pustaka

[5] Yunitasari, Y., Musdholifah, A., & Sari, A. K. (2019). Sarcasm Detection For Sentiment Analysis in Indonesian Tweets. IJCCS (Indonesian Journal of Computing and Cybernetics Systems), 13(1), 53. https://doi.org/10.22146/ijccs.41136

[6] Wijaya, Maranatha F. (2020). Analisis Kinerja Modifikasi Model Gabungan Long Short-Term Memory dan Convolution Neural Network untuk Analisis Sentimen Berbahasa Indonesia. Skripsi. FMIPA, Matematika. Universitas Indonesia.

[7] Yue, W., & Li, L. (2020, December 14). Sentiment analysis using word2vec-cnn-bilstm classification. 2020 7th International Conference on Social Network Analysis, Management and Security, SNAMS 2020. https://doi.org/10.1109/SNAMS52053.2020.9336549

[8] Chiu, J. P. C., & Nichols, E. (2016). Named Entity Recognition with Bidirectional LSTM-CNNs. https://doi.org/10.1162/tacl_a_00104

[9] Shen, Q., Wang, Z., & Sun, Y. (2017). Sentiment analysis of movie reviews based on CNN-BLSTM. IFIP Advances in Information and Communication Technology, 510, 164–171. https://doi.org/10.1007/978-3-319-68121-4_17

# Daftar Pustaka

[10] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning: The MIT Press, 2016, 800 pp, ISBN: 0262035618. (n.d.). Retrieved May 4, 2021, from https://www.researchgate.net/publication/320703571_Ian_Goodfellow_Yoshua_Bengio_and_Aaron_Courville_Deep_learning_The_MIT_Press_2016_800_pp_ISBN_0262035618

[11] Machine Learning : Kenali Perbedaannya Deep Learning dan Art. (n.d.). Retrieved May 4, 2021, from https://www.dqlab.id/machine-learning-kenali-perbedannya-deep-learning-dan-ai

[12] Trueman, T. E., J., A. K., P., N., & J., V. (2021). Attention-based C-BiLSTM for fake news detection. Applied Soft Computing, 110, 107600. https://doi.org/10.1016/j.asoc.2021.107600

[13] Rachmatullah, A. (2018). Optimasi Hyperparameter Pada Gradient Boosted Trees Menggunakan Bayesian. Skripsi. FASILKOM, Teknik Informatika. Universitas Sriwijaya.

# LAMPIRAN

```
## Import modul yang diperlukan
import numpy as np
import pandas as pd
import re
import tensorflow as tf
import keras
from sklearn.model_selection import train_test_split
from keras.layers import Bidirectional, LSTM
from keras.models import Sequential
from keras.preprocessing import text
from sklearn.svm import SVC
from sklearn import metrics
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
```

```
# Upload Data
quipper_df = pd.read_csv('https://raw.githubusercontent.com/Syamsyuriani/Scrapping_Data/main/Quipper-Data_labelled.csv')
```

```
# Menghapus duplikat
dt = review.drop_duplicates(subset=['review'], keep='last', inplace=False).reset_index()
df = dt.drop(['index'], axis=1)
```

```
# Menghapus duplikat
dt = review.drop_duplicates(subset=['review'], keep='last', inplace=False).reset_index()
df = dt.drop(['index'], axis=1)
```

```
negative = df[df['label']==-1].count()[0]
positive = df[df['label']==1].count()[0]
```

```
labels = ['Positive','Negative']
Category1 = [positive, negative]
plt.bar(labels, Category1, tick_label=labels, width=0.5, color=['coral', 'c'])
plt.xlabel('Kelas Sentimen')
plt.ylabel('Data')
plt.title('Diagram Bar Data Analisis Sentimen')
```

```
#PEMBERSIHAN(CLEANING DATA)
stopwords = pd.read_csv("https://raw.githubusercontent.com/listakurniawati/COVID-19-With-SVM/main/stopwords_id.csv?token=ARCQD7EZ55J4TUTAWYYLYOTAX3FTW")
stopwords = np.append(stopwords, "rt")

def clean_text(tweet):

    # Convert to lower case
    tweet = tweet.lower()
    # Clean www.* or https?://*
    tweet = re.sub('((www\.[^\s]+)|(https?://[^\s]+))','',tweet)
    # Clean @username
    tweet = re.sub('@[^\s]+','',tweet)
    #Remove punctuation
    tweet = re.sub(r'[^\w\s]',' ', tweet)
    #Replace #word with word
    tweet = re.sub(r'#([^\s]+)', r'\1', tweet)
    #Remove punctuation
    tweet = re.sub(r'[^\w\s]',' ', tweet)
    #Clean number
    tweet = re.sub(r'[\d-]', '', tweet)
    #Remove additional white spaces
    tweet = re.sub('[\s]+', ' ', tweet)
    #trim
    tweet = tweet.strip('\'"')
    # Clean per Words
    words = tweet.split()
    tokens=[]
    for ww in words:
        #split repeated word
        for w in re.split(r'[-/\s]\s*', ww):
            #replace two or more with two occurrences
            pattern = re.compile(r"(.)\1{1,}", re.DOTALL)
            w = pattern.sub(r"\1\1", w)
            #strip punctuation
            w = w.strip('\'"?,.')
            #check if the word cosists of two or more alphabets
            val = re.search(r"^[a-zA-Z][a-zA-Z][a-zA-Z]*$", w)
            #add tokens
            if(w in stopwords or val is None):
                continue
            else:
                tokens.append(w.lower())

    tweet = " ".join(tokens)
    return tweet
```

```
df['label'] = df['label'].replace(-1,0)
```

```
df['review'] = df['review'].map(lambda x: clean_text(x))
df = df[df['review'].apply(lambda x: len(x.split()) >=1)]
```

```
#tokenisasi
tokenizer = keras.preprocessing.text.Tokenizer()
tokenizer.fit_on_texts(df['review'])
```

```
#sequencing
sequences = tokenizer.texts_to_sequences(df['review'])
x = keras.preprocessing.sequence.pad_sequences(sequences, maxlen=80)
y = np.array((df['label']))
```

```
seq_length = x.shape[1]
print(seq_length)
vocab_size = len(tokenizer.index_word) + 1
print(vocab_size)
```

```
#pemisahan dataset dengan rasio train:test = 80:20
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size = 0.2, random_state = 42)
```

```
pip install -q -U keras-tuner
```

```
from tensorflow import keras
from kerastuner.tuners import BayesianOptimization

def cnn_bilstm(hp):

    ### Arsitektur

    #Input layer
    inputs = keras.layers.Input(shape=(seq_length,))

    #Embedding
    embedding = keras.layers.Embedding(input_dim = vocab_size,
                                       output_dim = hp.Choice('embedding_size', values = [64, 100, 128]))(inputs)
```

```
#Convolution layer
ngram_1 = keras.layers.Conv1D(filters = hp.Int('filters1',
                                               min_value = 200,
                                               max_value = 300,
                                               step = 50),
                              kernel_size = hp.Int('kernel_size1',
                                                   min_value = 3,
                                                   max_value = 5,
                                                   step = 1),
                              activation='relu',
                              kernel_regularizer = keras.regularizers.l2(hp.Choice('kernel_cnn1',
                                                                                   values = [0.0, 0.01])))(embedding)
ngram_2 = keras.layers.Conv1D(filters = hp.Int('filters2',
                                               min_value = 200,
                                               max_value = 300,
                                               step = 50),
                              kernel_size = hp.Int('kernel_size2',
                                                   min_value = 3,
                                                   max_value = 5,
                                                   step = 1),
                              activation='relu',
                              kernel_regularizer = keras.regularizers.l2(hp.Choice('kernel_cnn2',
                                                                                   values = [0.0, 0.01])))(embedding)
ngram_3 = keras.layers.Conv1D(filters = hp.Int('filters3',
                                               min_value = 200,
                                               max_value = 300,
                                               step = 50),
                              kernel_size = hp.Int('kernel_size3',
                                                   min_value = 3,
                                                   max_value = 5,
                                                   step = 1),
                              activation='relu',
                              kernel_regularizer = keras.regularizers.l2(hp.Choice('kernel_cnn3',
                                                                                   values = [0.0, 0.01])))(embedding)
ngram_4 = keras.layers.Conv1D(filters = hp.Int('filters4',
                                               min_value = 200,
                                               max_value = 300,
                                               step = 50),
                              kernel_size = hp.Int('kernel_size4',
                                                   min_value = 3,
                                                   max_value = 5,
                                                   step = 1),
                              activation='relu',
                              kernel_regularizer = keras.regularizers.l2(hp.Choice('kernel_cnn4',
                                                                                   values = [0.0, 0.01])))(embedding)
```

```
#Max Pooling layer
ngram_1 = keras.layers.GlobalMaxPooling1D()(ngram_1)
ngram_2 = keras.layers.GlobalMaxPooling1D()(ngram_2)
ngram_3 = keras.layers.GlobalMaxPooling1D()(ngram_3)
ngram_4 = keras.layers.GlobalMaxPooling1D()(ngram_4)
merged = keras.layers.Concatenate(axis=1)([ngram_1, ngram_2, ngram_3, ngram_4])
```

```
#BiLSTM layer
bilstm1 = keras.layers.Layer(LSTM(units = hp.Int('units1',
                                    min_value = 100,
                                    max_value = 200,
                                    step = 50),
                                kernel_regularizer=keras.regularizers.l2(hp.Choice('kernel_regularizer1',
                                                                                   values = [0.0, 0.01])),
                                recurrent_regularizer=keras.regularizers.l2(hp.Choice('rec_regularizer1',
                                                                                      values = [0.0, 0.01])),
                                return_sequences = True))(merged)
bilstm2 = keras.layers.Layer(LSTM(units = hp.Int('units2',
                                    min_value = 100,
                                    max_value = 200,
                                    step = 50),
                                kernel_regularizer=keras.regularizers.l2(hp.Choice('kernel_regularizer2',
                                                                                   values = [0.0, 0.01])),
                                recurrent_regularizer=keras.regularizers.l2(hp.Choice('rec_regularizer2',
                                                                                      values = [0.0, 0.01])),
                                return_sequences = True, go_backwards=True))(bilstm1)
```

```
#Dropout layer
lstm_out = keras.layers.Dropout(0.25)(bilstm2)
```

```
#Output layer
output = keras.layers.Dense(1, activation='sigmoid', kernel_regularizer=keras.regularizers.l2(hp.Choice('kernel_dense', values = [0.0, 0.01])))(lstm_out)
model = keras.models.Model(inputs=inputs, outputs=output)
```

```
model.compile(optimizer = keras.optimizers.Adam(
                          hp.Choice('learning_rate',
                                    values = [1e-2, 1e-3, 1e-4])),
                          loss='binary_crossentropy',
                          metrics=['accuracy'])

return model
```

```
tuner = BayesianOptimization(cnn_bilstm,
                             objective = 'val_accuracy',
                             max_trials = 10,
                             directory = 'Hasil',
                             project_name = 'Sentimen_CNN-BiLSTM')

tuner.search(x_train, y_train, batch_size=32, epochs = 10, validation_data = (x_test, y_test))

# Get the optimal hyperparameters
best_hps = tuner.get_best_hyperparameters()[0]

print('\nThe hyperparameter search is complete. \nembedding_size:', best_hps.get('embedding_size'),
      '\nfilters:', [best_hps.get('filters1'), best_hps.get('filters2'), best_hps.get('filters3'), best_hps.get('filters4')],
      '\nkernel_size:', [best_hps.get('kernel_size1'), best_hps.get('kernel_size2'), best_hps.get('kernel_size3'), best_hps.get('kernel_size4')],
      '\nkernel_cnn:', [best_hps.get('kernel_cnn1'), best_hps.get('kernel_cnn2'), best_hps.get('kernel_cnn3'), best_hps.get('kernel_cnn4')],
      '\nunit:', [best_hps.get('units1'), best_hps.get('units2')],
      '\nkernel_regularizer:', [best_hps.get('kernel_regularizer1'), best_hps.get('kernel_regularizer2')],
      '\nrec_regularizer:', [best_hps.get('rec_regularizer1'), best_hps.get('rec_regularizer2')],
      '\nkernel_dense:', best_hps.get('kernel_dense'),
      '\nLearning rate:', best_hps.get('learning_rate'))
```

```
from tensorflow.keras.utils import plot_model

model = tuner.hypermodel.build(best_hps)

#Plot Model
plot_model(model, to_file='model.png', show_shapes=True, show_layer_names=False, rankdir='TB')
```

```
# Summary Model
model.summary()
```

```
# Retrain model with the optimal hyperparameters
history = model.fit(x_train, y_train, batch_size=32, epochs = 10, validation_data = (x_test, y_test))
```

```
# Plot grafik loss dan accuracy
pd.DataFrame(history.history).plot(figsize=(8, 5))
plt.grid(True)
plt.gca().set_ylim(0,5)

plt.show()
```

```
## Evaluasi Model
test_loss, test_acc = model.evaluate(x_test, y_test)
print('Test accuracy:', test_acc)
```

```
## Penyimpanan Model
model.save('Data/model_CNN-BiLSTM_sentiment.h5')
```

```
# Upload data yang akan diprediksi
new_data = pd.read_csv('https://raw.githubusercontent.com/Syamsyuriani/Scrapping_Data/main/Quipper-Data.csv')
new_data
```

```
# Memuat kembali model
model = keras.models.load_model('Data/model_CNN-BiLSTM_sentiment.h5')
```

```
# Tokenisasi dan pad sequencing
tokenizer = keras.preprocessing.text.Tokenizer()
tokenizer.fit_on_texts(new_data['review'])
seq = tokenizer.texts_to_sequences(new_data['review'])
x_pred = keras.preprocessing.sequence.pad_sequences(seq, maxlen=80)
y_pred = model.predict(x_pred)

treshold = 0.5
for i in range(y_pred.shape[0]):
  if y_pred[i] > treshold:
    y_pred[i] = 1
  else:
    y_pred[i] = -1

new_data['label'] = y_pred
```

```
# Hasil data yang telah diberi label
pd.set_option("max_colwidth", 100)
pd.set_option("max_rows", None)
new_data
```

```
neg = new_data[new_data['label']==-1].count()[0]
pos = new_data[new_data['label']==1].count()[0]
```

```
labels = ['Positive','Negative']
Category1 = [str(pos),str(neg)]
plt.pie(Category1, labels=labels, autopct='%1.2f%%', colors=['coral', 'c'])
plt.xlabel('Kelas Sentimen')
plt.ylabel('Data')
plt.title('Diagram Lingkaran Data Analisis Sentimen')
```