Perbandingan Sent2vec TF-IDF Logistic Regression dan Word2vec CNN pada hasil Sentiment Analysis Youtube Comment

Aganda Maulana Dan Dyantono[1], Ricky Eka Putra[2]

[1,2] Jurusan Teknik Informatika/Teknik Informatika, Universitas Negeri Surabaya
[1]agandadyantono16051204038@mhs.unesa.ac.id
[2]rickyeka@unesa.ac.id

***Abstrak*—Kebutuhan akan internet untuk masyarakat Indonesia sangatlah masif, sehingga membuat ketergantungan akan penggunaan internet sangat tinggi untuk kebutuhan sehari-hari masyrakat Indonesia. Sosial media merupakan salah satu alasan untuk berhubungan langsung dengan internet, karena terdapat banyak informasi dan juga hiburan yang ada pada sosial media. *Youtube* merupakan salah satu *Platform* yang memiliki peringkat teratas untuk penggunaan sosial media di negara Indonesia.**

***Covid-19* merupakan musibah yang sangat besar untuk umat manusia, begitu juga dengan negara Indonesia yang ikut terkena dampak *Covid-19*. *Covid-19* memiliki dampak serius terhadap pendidikan yang ada di negara Indonesia, karena adanya pembatasan interaksi manusia secara langsung membuat Kementerian Pendidikan dan Kebudayaan (KEMENDIKBUD) mengambil keputusan untuk menerapkan pembelajaran secara tidak langsung atau disebut dengan *Daring*. *Daring* memiliki banyak opini dari masyarakat Indonesia, karena merupakan kebijakan baru untuk Pendidikan di negara Indonesia.**

Adapun penelitian ini dilakukan untuk memperoleh data dari *Scrapping Youtube Comment* untuk melakukan *Sentiment Analysis* dari opini masyarakat tentang kebijakan pendidikan di era pandemi yang diambil KEMENDIKBUD, dan juga untuk melakukan perbandingan model *Sent2vec TF-IDF* Logistic Regression (LR) dengan *Word2vec CBOW* Convolutional Neural Network (CNN) pada hasil *Sentiment Analysis* tersebut. Pada penelitian ini memiliki tahapan pengumpulan data yang diperoleh dari *Scrapping* data menggunakan Google Spreadsheet pada *Youtube Comment* di *Channel Youtube* Deddy Corbuzier yang berjudul "Nadiem, Kalau Bodoh Satu Generasi Gimana Bro?-Nadiem Makarim-Deddy Corbuzier Podcast", data *Scrapping* digunakan untuk melakukan *Sentiment Analysis* pada RStudio untuk mendapatkan penilaian pada data, data yang sudah dinilai akan digunakan untuk melakukan perbandingan model *Sent2vec TF-IDF LR* dengan *Word2vec CBOW CNN* pada Jupyter Notebook untuk mendapatkan nilai *Accuracy, Precision, Recall, F1 Score,* dan *ROC* pada masing-masing model sebagai acuan perbandingan antar model.

Dari hasil penelitian tersebut, mendapatkan kesimpulan untuk data *Scrapping Youtube Comment* mendapatkan 1 data dengan isi sebanyak 14.605 kalimat di dalamnya, pada *Sentiment Analysis* yang dilakukan mendapatkan hasil penilaian terhadap data yaitu 9.549 kalimat positif dan 5.056 kalimat negatif, untuk perbandingan model *Word2vec CBOW CNN* memiliki nilai lebih baik dari model *Sent2vec TF-IDF LR* dengan selisih hasil yaitu untuk *Accuracy* 4,09%, *Precision* 6,75%, *Recall* 0,06%, *F1 Score* 2,81%, dan *ROC* 0,2%.

***Kata Kunci*—scrapping data, sentiment analysis, sent2vec, tf-idf, logistic regression, word2vec, cbow, cnn**

I. PENDAHULUAN

Pengguna internet diseluruh dunia sangat besar dan internet sangat berdampak bagi kehidupan di dunia saat ini. Negara Indonesia memiliki wilayah yang sangat luas, juga berbanding lurus dengan populasi manusia dan juga menjadi salah satu penyumbang pengguna internaet terbesar di dunia. Berdasarkan survey penetrasi dan perilaku penggunaa internet APPJI pada 2019-Q2 2020, terdapat 196,7 juta pengguna internet di Indonesia setara 73,7% dari populasi, jumlah ini bertambah sekitar 25,5 juta pengguna dibandingkan tahun sebelumnya[1]. Sosial media merupakan faktor yang sangat penting terhadap pertumbuhan pengguna internet di Indonesia, banyak masyarakat Indonesia menggunakan sosial media untuk berbagai kebutuhan sehari-hari. We Are Social pada tahun ini menyebutkan pengguna sosial media di Indonesia mencapai 170 juta, angka pengguna aktif sosial media di Indonesia tumbuh sebesar 10 juta atau sekitar 6,3% dibandingkan Januari 2020[1]. Berikut adalah data dari pengguna sosial media di Indonesia berdasarkan survey We Are Social yaitu 88% pengguna Youtube, 83% Whatsapp, 81% pengguna Facebook, 80% pengguna Instagram, 59% pengguna Line, 52% pengguna Twitter, 47% pengguna Facebook Messenger, 38% pengguna Blackberry Messenger, 33% pengguna Linkdln, 29% pengguna Pinterest[1]. Dari data pengguna sosial media yang telah disebutkan sebelumnya, maka dapat dilihat bahwa Youtube merupakan Platform sosial media yang terbanyak digunakan masyarakat Indonesia.

Covid-19 adalah penyakit menular yang disebabkan oleh Sindrom pernapasan akut Corona Virus 2 (SARS-Cov-2), penyakit ini pertama kali diidentifikasi pada Desember 2019 di Wuhan Hubei China. Sejak *Covid-19* menyebar secara Global, telah mengakibatkan lebih dari 64.700 kematian, dan 246.000 orang telah pulih[2]. *Covid-19* merupakan bencana besar untuk umat manusia, begitu juga dengan negara Indonesia yang ikut terdampak, terutama untuk sektor pendidikan karena adanya pembatasan interaksi antar umat

manusia. Pada 15 Juni 2020, Kementerian Pendidikan dan kebudayaan (KEMENDIKBUD) mengumumkan Surat Keputusan Bersama (SKB) tentang panduan penyelenggaraan pembelajaran pada tahun ajaran dan tahun akademik baru di masa pandemi *Covid-19*, panduan tersebut akan berpengaruh untuk model pembelajaran di Indonesia pada saat tahun ajaran baru 2020/2021[3]. Pada 16 Juni 2020, UNESCO mencatat bahwa pandemi *Covid-19* menyebabkan hampir 1,1 miliar siswa yang belajar di 123 negara di dunia tidak dapat bersekolah/kuliah seperti sebelumnya, angka ini merupakan 62,3% dari jumlah mereka yang belajar di seluruh penjuru dunia, sebagai dampak penutupan Institusi Pendidikan[3]. karena adanya pembatasan interaksi manusia secara langsung membuat KEMENDIKBUD mengambil keputusan untuk menerapkan pembelajaran secara tidak langsung atau disebut dengan *Daring*. *Daring* memiliki banyak opini dari masyarakat Indonesia, karena merupakan kebijakan baru untuk Pendidikan di negara Indonesia.

Opinion Mining merupakan analisis opini dari suatu pola atau *mood* terhadap orang atau topik tertentu, hal seperti itu yang sering disebut *Sentiment*[4]. Opini masyarakat tentang topik yang ditentukan bisa diambil dari sosial media terutama pada *Platform Youtube,* karena terdapat banyak sekali opini masyarakat didalamnya. *Sentiment Analysis* digunakan untuk penilaian terhadap opini yang telah didapatkan, *Sentiment Analysis* akan mendapatkan data yang telah diberi nilai berupa positif, negatif, atau netral.

Penelitian ini dilakukan untuk memperoleh data dari *Scrapping Youtube Comment* untuk melakukan *Sentiment Analysis* dari opini masyarakat tentang kebijakan pendidikan di era pandemi yang diambil KEMENDIKBUD, dan juga untuk melakukan perbandingan model *Sent2vec TF-IDF* Logistic Regression (LR) dengan *Word2vec CBOW* Convolutional Neural Network (CNN) pada hasil *Sentiment Analysis* tersebut. Pada penelitian ini memiliki tahapan pengumpulan data yang diperoleh dari *Scrapping* data menggunakan Google Spreadsheet pada *Youtube Comment* di *Channel Youtube* Deddy Corbuzier yang berjudul "Nadiem, Kalau Bodoh Satu Generasi Gimana Bro?-Nadiem Makarim-Deddy Corbuzier Podcast", data *Scrapping* digunakan untuk melakukan *Sentiment Analysis* pada RStudio untuk mendapatkan penilaian pada data, data yang sudah dinilai akan digunakan untuk melakukan perbandingan model *Sent2vec TF-IDF LR* dengan *Word2vec CBOW CNN* pada Jupyter Notebook untuk mendapatkan nilai *Accuracy, Precision, Recall, F1 Score,* dan *ROC* pada masing-masing model sebagai acuan perbandingan antar model.

II. *DASAR TEORI*

A. Web Scrapping

Web Scrapping merupakan sebuah teknik yang digunakan untuk mengumpulkan data secara manual yaitu dengan melakukan *copy paste* data yang diinginkan maupun secara otomatis, yaitu dengan membuat sebuah model program atau kode yang dapat melakukan proses pengambilan data dari sebuah laman *Web*[5]. Sosial media merupakan salah satu tempat yang sering digunakan untuk melakukan *Scrapping* data pada laman *Web* karena sangat banyak terdapat berbagai macam informasi didalamnya.

B. Google Spreadsheet

Google Spreadsheet adalah aplikasi *Spreadsheet online* yang memungkinkan membuat dan memformat data serta dapat bekerja bersama orang lain secara online, manfaatnya yaitu dapat mengakses data dimanapun dan kapanpun, data tetap aman meski tersimpan secara online serta memiliki *real-time* data yang selalu *up to date* dan memiliki banyak fitur pengolahan data untuk memudahkan dalam pengoperasian[6]. Google Spreadsheet juga memiliki fitur untuk menginputkan kode program dan juga *resource API* yang sangat mendukung untuk melakukan *Scrapping Data* pada *Platform* sosial media yang diinginkan.

C. Sentiment Analysis

Sentiment Analysis adalah ekspresi subjektif, opini, perilaku, orientasi, dan emosi atau bahkan nada dalam teks yang sifatnya *unstructured* sekaligus *scattered* dan telah diolah dengan tepat yang menghasilkan informasi signifikan dan *meaningful*[7]. Secara garis besar Sentiment Analysis dibagi menjadi 2 bagian penting, yaitu *Coarse-grained sentiment analysis* dan *Fined-grained sentiment analysis*.

D. RStudio

RStudio merupakan suatu aplikasi yang berfungsu untuk menulis program dengan memakai bahasa R yang berbasis *open source* dan dapat digunakan di semua *OS* seperti Windows, Linux, dan Mac[8]. Pada RStudio sangat mendukung untuk melakukan proses *Sentiment Analysis* karena memiliki *Library* atau *Packages* yang mendukung dan banyak panduan penggunaan, karena sering digunakan untuk penelitian terdahulu.

E. Word Embedding

Word Embedding adalah sebuah fungsi parameter yang memetakan setiap kata kedalam Vector berdimensi tinggi[9]. *word embedding* dapat dibuat langsung dari *dataset* yang dimiliki atau menggunakan *pre-trained* pada *word embedding* yang telah tersedia untuk dilatih menggunakan dataset yang besar pada *domain* permasalahan tertentu yang dapat digunakan untuk permasalahan lain yang serupa dan harus di sesuaikan dengan *domain* kasus yang dimiliki.

F. Sent2vec

Sent2vec merupakan salah satu algoritma *word embedding* yang biasa digunakan untuk memetakan setiap kalimat kedalam vektor. Term Frequency - Inverse Document Frequency (TF-IDF) pertama kali dicetuskan oleh Salton dan Buckley pada tahun 1998 dan digunakan untuk kepentingan information retrieva, yang kemudian turut dimanfaatkan sebagai salah satu algoritma dalam metode *feature weighting* dalam *text mining*[10]. Rumus yang diperoleh yaitu :

TF-IDF = TF x IDF

TF = jumlah kemunculan term pada satu dokumen / jumlah seluruh term dalam dokumen

IDF = log x jumlah seluruh dokumen / jumlah dokumen suatu term yang muncul
Semakin sering sebuah fitur muncul dalam sebuah teks, maka semakin besar bobot yang akan didapatkan[10].

G. *Word2vec*

Word2vec merupakan salah satu algoritma *word embedding* yang memetakan setiap kata dalam teks ke dalam vektor[9]. *Word2vec* adalah salah satu aplikasi *unsupervised learning* menggunakan *neural network* yang terdiri dari sebuah *hidden layer* dan *fully connected layer*, dimensi dari matriks bobot pada setiap layer adalah jumlah dengan kata korpus dikalikan dengan jumlah *hidden neuron* pada *hidden layernya*. Matriks bobot pada *hidden layer* dari model yang telah dilatih digunakan untuk mentransformasikan kata kedalam vektor, matriks bobot ini seperti *lookup table*, dimana setiap baris mewakili setiap kata dan kolom mewakili vektor dari kata tersebut, terdapat dua algoritma *word2vec* yaitu Continuous Bag Of Word (CBOW) dan Skip Gram(SG).

H. *Convolutional Neural Network (CNN)*

Metode Convolutional Neural Network (CNN) adalah bagian dari *deep learning natural language processing* dalam sejumlah permasalahan klasifikasi yang mampu secara efisien menangkap representasi bermakna dari kalimat seperti pada klasifikasi dan pemodelan bahasa, analisis sentimen, hingga ekstraksi informasi[9].

I. *Logistic Regression (LR)*

Regresi Logistik adalah salah satu dari banyak metode pembelajaran mesin yang bekerja dengan mengambil input dan mengalikan nilai input dengan bobot dan juga berguna sebagai *classifier* untuk mempelajari fitur dari input yang paling berguna untuk membedakan antara berbagai kelas[10].

J. *Jupyter Notebook*

Jupyter Notebook adalah suatu aplikasi berbasis web yang digunakan untuk menulis program dengan pendekatan *console*, komputasi secara interaktif, bias di akses dari banyak c komputer klien[11].

K. *Penelitian Terdahulu*

Untuk melakukan suatu penelitian sangat dibutuhkan rujukan atau hasil dari penelitian terdahulu yang berkaitan untuk penelitian ini, untuk mendapatkan hasil yang lebih maksimal, sehingga penelitian yang akan dilakukan mendapatkan hasil memuaskan. Tabel 1 adalah contoh beberapa penelitian terdahulu tentang pembahasan dan hasil penelitian.

TABEL I
PENELITIAN TERDAHULU

No	Penulis	Masalah	Metode	Hasil
1.	(Nasichuddin et al., 2018)	Peningkatan performa untuk analisis sentiment dengan CNN	Membandingkan pra pelatihan Word2vec dan one hot encoding	Metode CNN dengan model Word2vec mendapatkan akurasi terbaik yaitu 81,10%.
2.	(Anindyati et al., 2019)	Pengoptimalan deeplearning untuk deteksi Cyberbullying	Membandingkan hasil klasifikasi CNN, LSTM, dan BLSTM	Metode CNN menghasilkan akurasi terbaik yaitu 91,03%.
3.	(Kiran et al., 2020)	Perbandingan arsitektur deeplearning untuk drug reviews	Membandingkan metode CNN, RF, SVM, GICF	Akurasi dengan metode CNN yaitu 93,1% dan itu sangat baik.
4.	(Muhammad Arief Rahman et al., 2019)	Opini public tentang operator telekomunikasi di instagram	Word embedding GloVe dan CNN	Menghasilkan presisi 95,80%, recall 88,12%, dan f-one 91,62 %.
5.	(Maulana & Rochmawati, 2020)	Opinion mining terhadap pemberitaan corona (Instagram)	Metode CNN dan Word embedding tf idf	Hasil yang di dapat yaitu presisi 96%, akurasi 88%.

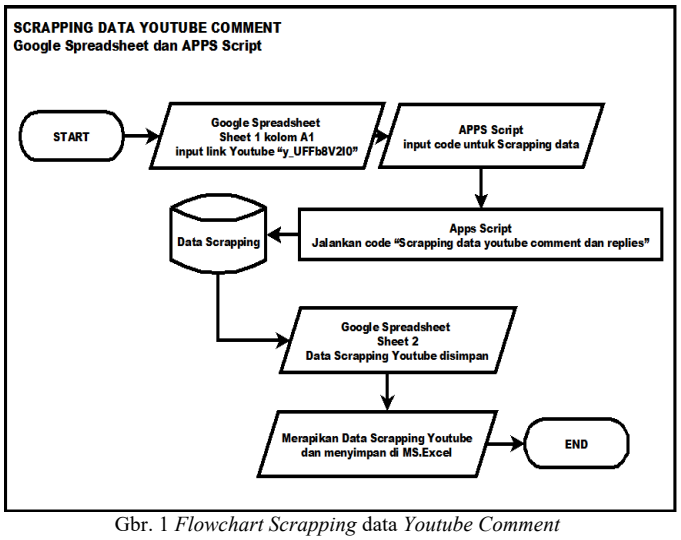
Pada penelitian terdahulu tentang *word embedding* dengan banyak metode, dari berbagai metode yang digunakan dapat terlihat untuk metode cnn mendapatkan hasil akurasi mencapai 90%, hasil tersebut sangat baik digunakan sebagai acuan untuk penelitian ini.

III. *METODOLOGI PENELITIAN*

Penelitian ini memiliki tahapan pengumpulan data +-15.000 data yang diperoleh dari *Scrapping* data menggunakan Google Spreadsheet pada *Youtube Comment* di *Channel Youtube* Deddy Corbuzier yang berjudul "Nadiem, Kalau Bodoh Satu Generasi Gimana Bro?-Nadiem Makarim-Deddy Corbuzier Podcast", data *Scrapping* digunakan untuk melakukan *Sentiment Analysis* pada RStudio untuk mendapatkan nilai positif dan negatif pada data, data yang sudah dinilai akan digunakan untuk melakukan perbandingan model *Sent2vec* TF-IDF LR dengan *Word2vec* CBOW CNN pada Jupyter Notebook untuk mendapatkan nilai *Accuracy, Precision, Recall, F1 Score,* dan *ROC* pada masing-masing model sebagai acuan perbandingan antar model.

A. *Pengumpulan data*

Data pada penelitian ini menggunakan data hasil Scrapping pada Comment Youtube di *Channel Youtube* Deddy Corbuzier yang berjudul "Nadiem, Kalau Bodoh Satu Generasi Gimana Bro?-Nadiem Makarim-Deddy Corbuzier Podcast".



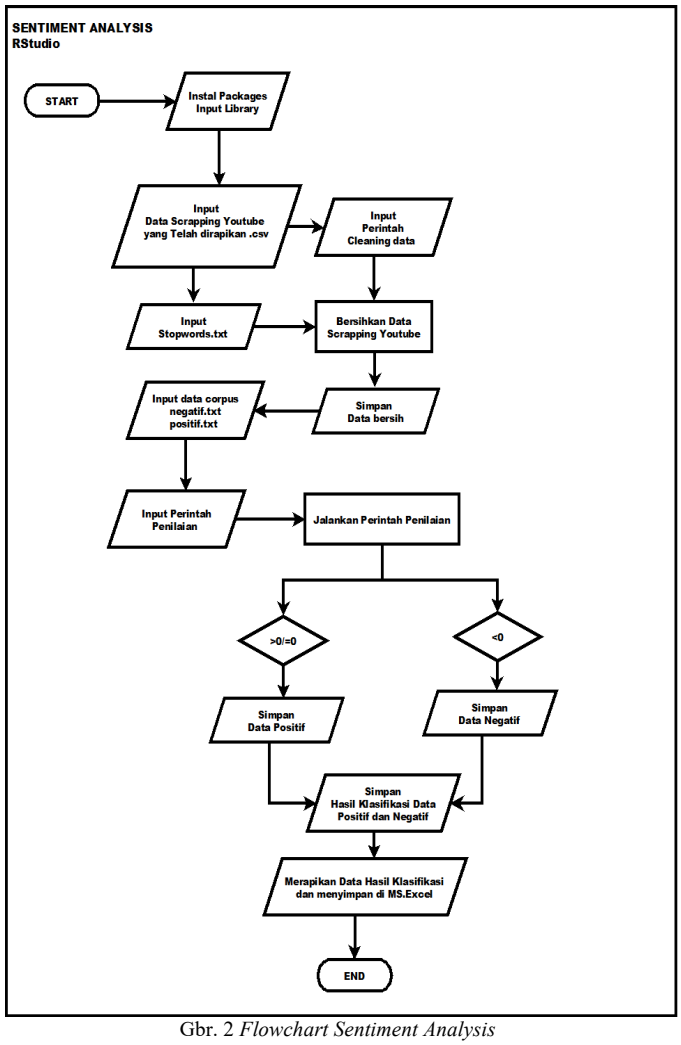
Gbr. 1 *Flowchart Scrapping* data *Youtube Comment*

Pada Gbr. 1 dapat dilihat untuk *Flowchart Scrapping* data *Youtube Comment* dengan tahapan melakukan *input link key Youtube* pada Google Spreadsheet Sheet 1 pada kolom A1, selanjutnya *input* perintah *Scrapping* pada Apps Script dengan menambahkan *resource API Youtube*, setelah menjalankan

program maka data akan ditampilkan pada Google Spreadsheet Sheet 2, data akan dirapikan dan disimpan pada Ms.excel.

B. *Labelling data*

Data hasil *Scrapping* yang telah dirapikan dan disimpan pada Ms.excel dilanjutkan dengan pelabelan untuk mendapatkan nilai sentiment positif(1) dan negatif(0) yang akan dilakukan di RStudio.

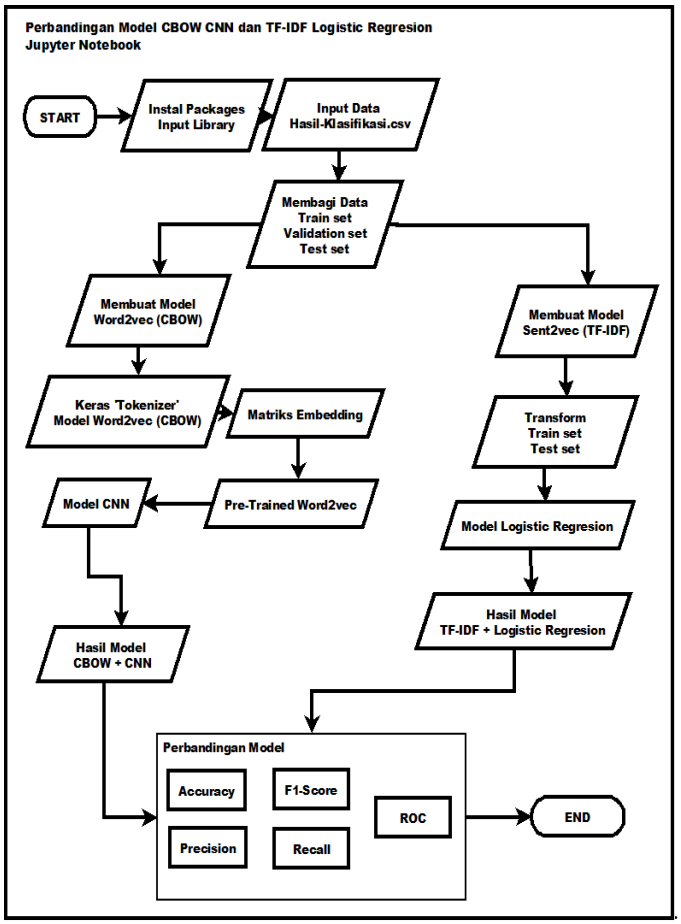


Gbr. 2 *Flowchart Sentiment Analysis*

Pada Gbr. 2 dapat dilihat untuk *Flowchart Sentiment Analysis* yang dilakukan pada aplikasi RStudio, tahapan yang dilakukan pertama *install packages* dan *library* kebutuhan, dilanjutkan dengan *input* data hasil *scrapping* yang telah disimpan pada Ms.excel, *input* stopwords.txt yang telah didefinisikan sendiri dan *input* perintah *cleaning* untuk menghapuskan kata yang tidak perlu, setelah itu menyimpan data bersih.csv, *input* perintah penilaian untuk melakukan penilaian menggunakan data positif.txt dan negatif.txt yang didefinisikan sendiri, perintah program berjalan jika total nilai >0/=0 maka data akan diberi label positif(1) dan jika nilai <0 maka akan diberi label negatif(0), setelah selesai menjalankan perintah penilaian maka data hasil penilaian akan disimpan pada Ms.excel, sebagai data untuk melakukan proses selanjutnya yaitu perbandingan model yang akan dilakukan di Jupyter Notebook.

C. *Implementasi dan Hasil Model*

Implementasi pada model akan dilakukan di Jupyter Notebook untuk melakukan perbandingan model *Sent2vec* TF-IDF LR dengan *Word2vec* CBOW CNN pada hasil data yang telah diberi label positif(1) dan negatif(0).



Gbr. 3 *Flowchart* Implementasi dan Hasil model

Pada Gbr. 3 dapat dilihat untuk *Flowchart* Implementasi dan Hasil model pada Jupyter Notebook dengan tahapan *install packages* dan *input library* kebutuhan, *input* data hasil *sentiment* yang telah diberi label, dilanjutkan dengan membagi data set, untuk Sent2vec TF-IDF LR dilakukan Transform data dilanjutkan dengan menentukan model LR dan menjalankan model TF-IDF LR untuk memperoleh hasil yang di inginkan, untuk Word2vec CNN membuat model Word2vec CBOW SG, *Tokenizer*, melakukan *word embedding* untuk proses *pre-trained* model *Word2vec* dilanjutkan dengan membuat model CNN dan menggabungkan hasil model Word2vec CNN untuk mendapatkan hasil yang diinginkan. Hasil yang diinginkan adalah nilai Accuracy, Precision, F1-Score, Recall, dan ROC dengan rumus sebagai berikut :

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

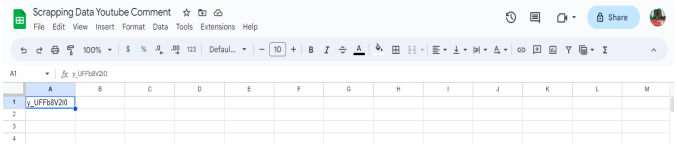
$$F1\ Score = 2 * \frac{precision * recall}{precision + recall} \quad (4)$$

Dari rumus yang telah di definisikan diatas, maka nilai dapat diketahui sebagai acuan untuk perbandingan model yang telah dibuat.

IV. HASIL DAN PEMBAHASAN

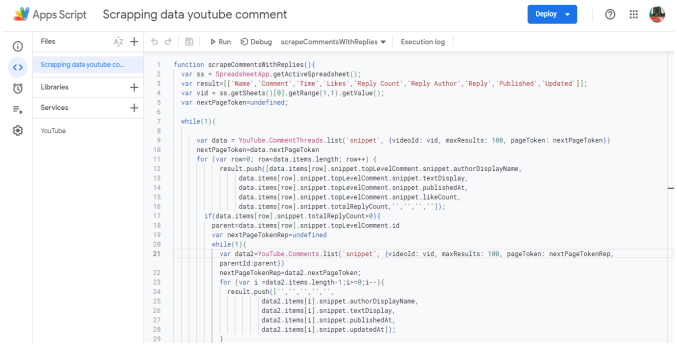
A. *Pengumpulan data*

Setelah mendapatkan tujuan penelitian, maka dilakukan tahapan untuk mencari dan mengumpulkan data, pengumpulan data dilakukan dengan cara *Scrapping* pada *Youtube Comment* menggunakan Google Spreadsheet pada Sheet 1 kolom A1 untuk *input* sumber *key link Youtube* yaitu **y_UFFb8V2I0**. Dibawah ini merupakan tampilan pada Google Spreadsheet Sheet 1 yang telah dibuat.



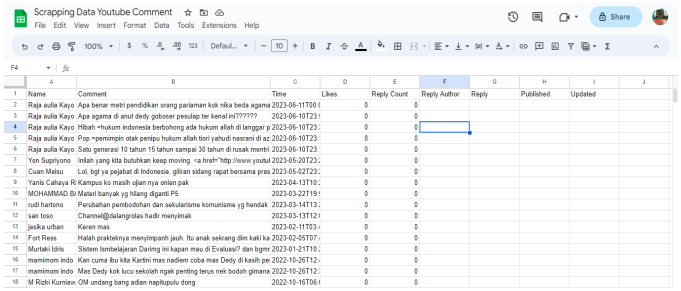
Gbr. 4 Google Spreadsheet Sheet 1

Selanjutnya menggunakan Apps Script pada Google Spreadsheet untuk menuliskan kode program atau perintah pengambilan data yaitu untuk mengambil Nama, Comment, Time, Likes, Replay Count, Replay Author, Replay, Published, Updated dari Youtube yang memiliki *key link Youtube* **y_UFFb8V2I0,** dengan menambahkan Youtube API pada bagian services untuk mendapatkan izin *Scrapping* data pada Youtube. Dibawah ini merupakan tampilan untuk Apps Script pada Google Spreadsheet.



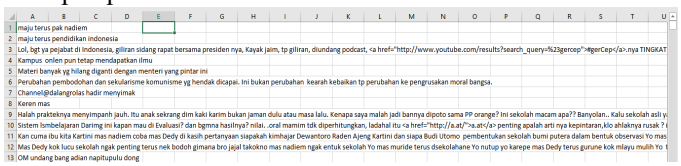
Gbr. 5 Apps Script pada Google Spreadsheet

Setelah menjalankan kode program Apps Script pada Google Spreadsheet maka data sesuai perintah akan muncul pada Google Spreadsheet Sheet 2. Dibawah ini merupakan tampilan Google Spreadsheet Sheet 2.



Gbr. 6 Google Spreadsheet Sheet 2

Selanjutnya merapikan data yaitu mengambil kolom Comment dan kolom Reply untuk digabungkan jadi satu kolom dan disimpan pada Ms.excel. Dibawah ini merupakan tampilan data hasil *Scrapping* yang telah dirapikan dan disimpan pada Ms.excel.



Gbr. 7 Data rapi hasil *Scrapping*

B. *Labelling data*

Data yang diperoleh dari proses pengumpulan data dengan cara *Scrapping* pada *Youtube Comment* akan melalui proses *Labelling* di RStudio. Untuk melakukan proses *Labelling* data, maka diperlukan *install Packages* untuk *library* yaitu tm, snowbalic, wordcloud, wordcloud2, RcolorBrewer, stringr, Rcurl,corpus,dan plyr. Selanjutnya *input* data rapi hasil *Scrapping* untuk dilakukan proses *Cleaning* data atau menghapus data dari kata yang tidak diperlukan, dengan *input* kode program *Cleaning* data dan dengan menggunakan Stopwords.txt yang telah didefinisikan sendiri sebagai kata yang tidak diperlukan untuk melakukan tahapan penilaian. Dibawah ini adalah kode program untuk *Cleaning* data.

```
setwd('E:\\skripsiku')
docs<-readLines("comment deddy fix.csv")

docs <- Corpus(VectorSource(docs))
inspect(docs)
toSpace <- content_transformer(function (x , pattern ) gsub(pattern, " ", x))
docs <- tm_map(docs, toSpace, "/")
docs <- tm_map(docs, toSpace, "@")
docs <- tm_map(docs, toSpace, "\\|")
docs <- tm_map(docs, content_transformer(tolower))
docs <- tm_map(docs, toSpace, "[[:punct:]]")
docs <- tm_map(docs, toSpace, "[[:digit:]]")

myStopwords = readLines('D:\\kuliah\\semester 6\\big data\\fix\\stopwords.txt')
docs <- tm_map(docs, removeWords, myStopwords)
docs <- tm_map(docs, removeWords, c("sampah"))
docs <- tm_map(docs, stripWhitespace)
removeURL <- function(x) gsub("http[[:alnum:]]*", " ", x)
docs <- tm_map(docs, removeURL)
```

Gbr. 8 Kode program *Cleaning* data

Setelah melakukan proses *Cleaning* data maka akan dilanjutkan dengan membuat *term document matrix* untuk menyimpan data hasil dari proses *Cleaning* dengan format csv. Dibawah in merupakan kode program untuk melakukan *term document matrix* dan menyimpan data dengan format csv sebagai data bersih hasil dari proses *Cleaning*.

```
dtm <- TermDocumentMatrix(docs)
m <- as.matrix(dtm)
v <- sort(rowSums(m),decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d, 15)

dataframe<-data.frame(text=unlist(sapply(docs, `[`)), stringsAsFactors=F)

write.csv(dataframe, "E:\\skripsiku\\databersih.csv")
save.image()


data_clean <- read.csv('E:\\skripsiku\\databersih.csv')
str(data_clean)
```

Gbr. 9 Kode program term document matrix dan simpan data bersih

Selanjutnya untuk tahap *Labelling* atau memberikan nilai pada data yaitu untuk melakukan penilaian menggunakan data positif.txt dan negatif.txt yang didefinisikan sendiri, perintah program berjalan jika total nilai >0/=0 maka data akan diberi label positif(1) dan jika nilai <0 maka akan diberi label negatif(0). Dibawah ini adalah untuk kode program penilaian pada data bersih.

```
data_clean <- read.csv('E:\\skripsiku\\databersih.csv')
str(data_clean)

positif <- scan("D:\\kuliah\\semester 6\\big data\\fix\\positif.txt",what="character",comment.char=";")
negatif <- scan("D:\\kuliah\\semester 6\\big data\\fix\\negatif.txt",what="character",comment.char=";")
score.sentiment = function(kalimat2, kata.positif, kata.negatif, .progress='none')
{
  require(plyr)
  require(stringr)
  scores = laply(kalimat2, function(kalimat, kata.positif, kata.negatif) {
    kalimat = gsub('[[:punct:]]', '', kalimat)
    kalimat = gsub('[[:cntrl:]]', '', kalimat)
    kalimat = gsub('\\d+', '', kalimat)
    kalimat = tolower(kalimat)

    list.kata = str_split(kalimat, '\\s+')
    kata2 = unlist(list.kata)
    positif.matches = match(kata2, kata.positif)
    negatif.matches = match(kata2, kata.negatif)
    positif.matches = !is.na(positif.matches)
    negatif.matches = !is.na(negatif.matches)
    score = sum(positif.matches) - (sum(negatif.matches))
    if (score == 0){
      score = c("1")
    }
    else if (score < 0){
      score = c("0")
    }
    else if (score > 0){
      score = c("1")
    }
    return(score)
  }, kata.positif, kata.negatif, .progress=.progress )
  scores.df = data.frame(score=scores, text=kalimat2)
  return(scores.df)
}
```

Gbr. 10 Kode program penilaian pada data bersih

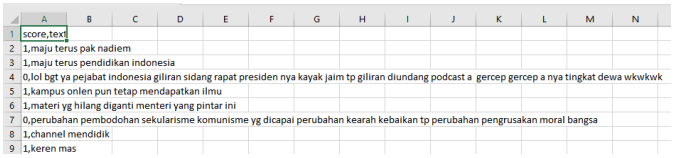
Setelah melakukan proses penilaian pada data bersih maka data hasil penilaian akan disimpan dengan format csv, untuk melihat tampilan visual dari data hasil penilaian maka akan dilakukan pembuatan barplot dari data hasil penilaian. Dibawah ini adalah kode program untuk simpan data hasil penilaian dan membuat barplot dari data hasil penilaian.

```
hasil2 = score.sentiment(data_clean$text, kata.positif, kata.negatif)
View(hasil2)
write.csv(hasil2,file = 'E:\\skripsiku\\hasil-klasifikasi.csv')

plot(hasil2$score,
     xlab= 'klasifikasi',
     ylab= 'jumlah comment',
     main= "Barplot sentiment",
     col=c('red','green'))
```

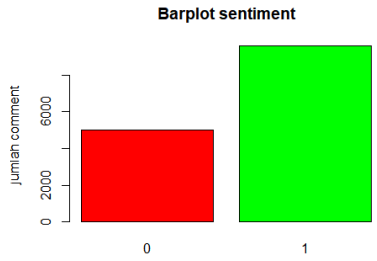
Gbr. 11 Kode Program simpan data hasil penilaian dan membuat barplot

Data hasil *labelling* yang didapatkan dari proses penilaian adalah sebanyak 14.604 data dengan label positif(1) sebanyak 9.459 data dan untuk label negatif(0) sebanyak 5.056 data. Data tersebut akan disimpan dalam format csv untuk proses selanjutnya, yaitu perbandingan model pada Jupyter Notebook. Dibawah ini merupakan tampilan beberapa contoh dari data hasil *labelling* dan juga tampilan visual dari barplot yang telah dibuat.



	A
1	score,text
2	1,maju terus pak nadiem
3	1,maju terus pendidikan indonesia
4	0,lol bgt ya pejabat indonesia giliran sidang rapat presiden nya kayak jaim tp giliran diundang podcast a gercep gercep a nya tingkat dewa wkwkwk
5	1,kampus onlen pun tetap mendapatkan ilmu
6	1,materi yg hilang diganti menteri yang pintar ini
7	0,perubahan pembodohan sekularisme komunisme yg dicapai perubahan kearah kebaikan tp perubahan pengrusakan moral bangsa
8	1,channel mendidik
9	1,keren mas

Gbr. 12 Data hasil *labelling*



Gbr. 13 Barplot dari data hasil labelling

C. *Implementasi dan Hasil model*

Pada penelitian ini untuk tahap implementasi model akan dilakukan di Jupyter Notebook untuk melakukan perbandingan model Sent2vec TF-IDF LR dengan Word2vec CBOW CNN pada hasil data yang telah diberi label positif(1) dan negatif(0). Selanjutnya melakukan *install Packages* dan juga *input library* yang dibutuhkan, dilanjutkan dengan *input* data hasil *labelling* pada Jupyter Notebook. Dibawah ini adalah kode program untuk *input* data hasil *labelling* dan tampilan data.

```
csv = 'hasil-klasifikasi.csv'
my_df = pd.read_csv(csv)
my_df.head()
```

	score	text
0	1	maju terus pak nadiem
1	1	maju terus pendidikan indonesia
2	0	lol bgt ya pejabat indonesia giliran sidang ra...
3	1	kampus onlen pun tetap mendapatkan ilmu
4	1	materi yg hilang diganti menteri yang pintar ini

Gbr. 14 kode program untuk *input* data hasil *labelling* dan tampilan data

Selanjutnya melakukan split data untuk memisahkan data testing, training dan validation. Dibawah ini adalah kode program split data.

```
from sklearn.feature_extraction.text import TfidfVectorizer
tvec = TfidfVectorizer(max_features=100000,ngram_range=(1, 3))
tvec.fit(x_train)
```

```
TfidfVectorizer(max_features=100000, ngram_range=(1, 3))
```

```
x_train_tfidf = tvec.transform(x_train)
x_test_tfidf = tvec.transform(x_test)
```

```
x_validation_tfidf = tvec.transform(x_validation).toarray()
```

Gbr. 15 Kode program spli data

Pada tahap selanjutnya yaitu membuat model untuk Sent2vec TF-IDF dengan cara import TF-IDFVectorizer dengan max_features = 10000, ngram_range = (1,3), dilanjutkan dengan melakukan transform dengan melakukan

tvec.transform pada data training, testing, dan validation. Dibawah ini adalah kode program untuk *modelling* Sent2vec TF-IDF.

```
from sklearn.feature_extraction.text import TfidfVectorizer
tvec = TfidfVectorizer(max_features=100000,ngram_range=(1, 3))
tvec.fit(x_train)
```

```
TfidfVectorizer(max_features=100000, ngram_range=(1, 3))
```

```
x_train_tfidf = tvec.transform(x_train)
x_test_tfidf = tvec.transform(x_test)
```

```
x_validation_tfidf = tvec.transform(x_validation).toarray()
```

Gbr. 16 Kode program *modelling* Sent2vec TF-IDF

Selanjutnya melakukan *modelling* untuk LR dengan cara import LogisticRegression pada sklearn.linier dan melakukan fit pada data training yang telah di transform dan data training nilai. Dibawah ini adalah kode program untuk *modelling* LR.

```
from sklearn.linear_model import LogisticRegression
```

```
lr_with_tfidf = LogisticRegression()
lr_with_tfidf.fit(x_train_tfidf,y_train)
```

```
LogisticRegression()
```

Gbr. 17 Kode program *modelling* LR

Untuk mendapatkan nilai dari *modelling* Sent2vec TF-IDF LR akan dilakukan predict pada data testing TF-IDF dilanjutkan confusion matrix pada sklearn.metrics untuk mendapatkan nilai yang diinginkan. Dibawah ini adalah kode program untuk predict dan confusion matrx.

```
y_predict_tfidf = lr_with_tfidf.predict(x_test_tfidf)
```

```
from sklearn.metrics import (
    confusion_matrix,
    accuracy_score,
    precision_score,
    recall_score,
    f1_score
)

conf_matrix = confusion_matrix(y_test, y_predict_tfidf)
conf_matrix
```

```
array([[39, 12],
       [11, 85]], dtype=int64)
```

Gbr. 18 Kode program predict dan confusion matrix

Selanjutnya untuk menampilkan hasil accuracy, precision, recall dan f1-score dari *modelling* Sent2vec TF-IDF LR akan dilakukan dengan rumus pada kode program yang ada dibawah ini.

```
precision = precision_score(y_test, y_predict_tfidf)
recall = recall_score(y_test, y_predict_tfidf)
f1score = f1_score(y_test, y_predict_tfidf)
accuracy = accuracy_score(y_test, y_predict_tfidf)

print(f"Accuracy = {accuracy}")
print(f"Precision = {precision}")
print(f"Recall = {recall}")
print(f"F1 Score = {f1score}")
```

Gbr. 19 Kode program rumus hasil

Untuk nilai dari hasil *modelling* Sent2vec TF-IDF LR dapat dilihat dibawah ini.

```
Accuracy = 0.8435374149659864
Precision = 0.8762886597938144
Recall = 0.8854166666666666
F1 Score = 0.8808290155440415
```

Gbr. 20 hasil *modelling* Sent2vec TF-IDF LR

Pada tahap selanjutnya yaitu melakukan modelling untuk Word2vec dengan melakukan split data menggunakan *library* taggeddocument dilanjutkan dengan membuat model yaitu sg =0, epochs =100, negative =5, window =2, min_count =2, alpha =0,065, min_alpha =0,065 dilanjutkan dengan cbow dan sg dengan range =30, epochs =1, alpha =0.002, min_alpha =0.002. Menyimpan data model Word2vec yang dilatih, dan memuat dengan fungsi "KeyedVectors" di Gensim. Dibawah ini adalah kode program untuk simpan dan memuat data *modelling* Word2vec.

```
model_ug_cbow.save('w2v_model_ug_cbow.word2vec')
model_ug_sg.save('w2v_model_ug_sg.word2vec')
```

```
from gensim.models import KeyedVectors
model_ug_cbow = KeyedVectors.load('w2v_model_ug_cbow.word2vec')
model_ug_sg = KeyedVectors.load('w2v_model_ug_sg.word2vec')
```

Gbr. 21 Kode program simpan dan muat data *modelling* Word2vec

Selanjutnya melakukan Tokenizer dan membuat layer word embedding dengan kode program dibawah ini.

```
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences

tokenizer = Tokenizer(num_words=100000)
tokenizer.fit_on_texts(x_train)

sequences = tokenizer.texts_to_sequences(x_train)
x_train_seq = pad_sequences(sequences, maxlen=45)

sequences_val = tokenizer.texts_to_sequences(x_validation)
x_val_seq = pad_sequences(sequences_val, maxlen=45)
```

Gbr. 22 Kode program tokenizer

```
num_words = 100000
embedding_matrix = np.zeros((num_words, 200))
for word, i in tokenizer.word_index.items():
    if i >= num_words:
        continue
    embedding_vector = embeddings_index.get(word)
    if embedding_vector is not None:
        embedding_matrix[i] = embedding_vector
```

Gbr. 23 Kode program layer word embedding

Setelah semua data telah siap maka akan dilakukan pembuatan model pre-train Word2vec untuk mendapatkan nilai yang di inginkan. Dibawah ini adalah kode program dan hasil untuk model pre-train Word2vec.

```
seed = 7

from keras.models import Sequential
from keras.layers import Dense, Dropout
from keras.layers import Flatten
from keras.layers.embeddings import Embedding
```

```
model_ptw2v = Sequential()
e = Embedding(100000, 200, weights=[embedding_matrix], input_length=45, trainable=False)
model_ptw2v.add(e)
model_ptw2v.add(Flatten())
model_ptw2v.add(Dense(256, activation='relu'))
model_ptw2v.add(Dense(1, activation='sigmoid'))
model_ptw2v.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model_ptw2v.fit(x_train_seq, y_train, validation_data=(x_val_seq, y_validation), epochs=5, batch_size=32, verbose=2)
```

Gbr. 24 Kode program model pre-train Word2vec 1

TABEL 2
HASIL PRE-TRAIN WORD2VEC 1

Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	0,5087	0,7507	0,4766	0,7740
2	0,3189	0,8685	0,4772	0,7740
3	0,1542	0,9554	0,5314	0,7945
4	0,0775	0,9840	0,6517	0,7945
5	0,0497	0,9901	0,7244	0,7945

```
model_ptw2v = Sequential()
e = Embedding(100000, 200, input_length=45)
model_ptw2v.add(e)
model_ptw2v.add(Flatten())
model_ptw2v.add(Dense(256, activation='relu'))
model_ptw2v.add(Dense(1, activation='sigmoid'))
model_ptw2v.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model_ptw2v.fit(x_train_seq, y_train, validation_data=(x_val_seq, y_validation), epochs=5, batch_size=32, verbose=2)
```

Gbr. 25 Kode program model pre-train Word2vec 2

TABEL 3
HASIL PRE-TRAIN WORD2VEC 2

Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	0,3710	0,8341	0,2432	0,9041
2	0,0682	0,9777	0,2878	0,8904
3	0,0114	0,9967	0,3632	0,8973
4	0,0074	0,9985	0,3493	0,8836
5	0,0025	0,9993	0,3996	0,8767

```
model_ptw2v = Sequential()
e = Embedding(100000, 200, weights=[embedding_matrix], input_length=45, trainable=True)
model_ptw2v.add(e)
model_ptw2v.add(Flatten())
model_ptw2v.add(Dense(256, activation='relu'))
model_ptw2v.add(Dense(1, activation='sigmoid'))
model_ptw2v.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model_ptw2v.fit(x_train_seq, y_train, validation_data=(x_val_seq, y_validation), epochs=5, batch_size=32, verbose=2)
```

Gbr. 26 Kode program model pre-train Word2vec 3

TABEL 4
HASIL PRE-TRAIN WORD2VEC 3

Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	0,4973	0,7654	0,4163	0,8151
2	0,2279	0,9112	0,3746	0,8219
3	0,0768	0,9795	0,5074	0,8219
4	0,0368	0,9932	0,4965	0,8219
5	0,0224	0,9961	0,5152	0,8425

Hasil dari pre-train model Word2vec menunjukkan seberapa bagus persiapan data yang telah disiapkan untuk model Word2vec, dilihat dari hasil pre-train yang mendapatkan 99% training accuracy dan 90% validation accuracy yang berada pada pre-train model Word2vec.

Setelah mendapatkan nilai dari pre-train yang baik maka akan dilanjutkan untuk melihat structure test dan akan dilanjutkan dengan membuat model Word2vec CNN agar mendapatkan hasil yang diinginkan. Dibawah ini adalah proses untuk melihat structure test dan model untuk Word2vec CNN beserta hasil yang dapat dilihat pada tabel di masing-masing model.

```
structure_test = Sequential()
e = Embedding(100000, 200, input_length=45)
structure_test.add(e)
structure_test.add(Conv1D(filters=100, kernel_size=2, padding='valid', activation='relu', strides=1))
structure_test.add(GlobalMaxPooling1D())
structure_test.summary()
```

```
Model: "sequential_18"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_19 (Embedding)    (None, 45, 200)           20000000

 conv1d_12 (Conv1D)          (None, 44, 100)           40100

 global_max_pooling1d_9 (Glo  (None, 100)              0
 balMaxPooling1D)

=================================================================
Total params: 20,040,100
Trainable params: 20,040,100
Non-trainable params: 0
_________________________________________________________________
```

Gbr. 27 Structure test

```
model_cnn_01 = Sequential()
e = Embedding(100000, 200, weights=[embedding_matrix], input_length=45, trainable=False)
model_cnn_01.add(e)
model_cnn_01.add(Conv1D(filters=100, kernel_size=2, padding='valid', activation='relu', strides=1))
model_cnn_01.add(GlobalMaxPooling1D())
model_cnn_01.add(Dense(256, activation='relu'))
model_cnn_01.add(Dense(1, activation='sigmoid'))
model_cnn_01.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model_cnn_01.fit(x_train_seq, y_train, validation_data=(x_val_seq, y_validation), epochs=5, batch_size=32, verbose=2)
```

Gbr. 28 Model training Word2vec CNN 1

TABEL 5
HASIL TRAINING WORD2VEC CNN 1

Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	0,4374	0,7946	0,3562	0,8151
2	0,2884	0,8793	0,3054	0,8493
3	0,2030	0,9237	0,5377	0,8082
4	0,1450	0,9480	0,3602	0,8699
5	0,0938	0,9671	0,4529	0,8767

```
model_cnn_02 = Sequential()
e = Embedding(100000, 200, input_length=45)
model_cnn_02.add(e)
model_cnn_02.add(Conv1D(filters=100, kernel_size=2, padding='valid', activation='relu', strides=1))
model_cnn_02.add(GlobalMaxPooling1D())
model_cnn_02.add(Dense(256, activation='relu'))
model_cnn_02.add(Dense(1, activation='sigmoid'))
model_cnn_02.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model_cnn_02.fit(x_train_seq, y_train, validation_data=(x_val_seq, y_validation), epochs=5, batch_size=32, verbose=2)
```

Gbr. 29 Model training Word2vec CNN 2

TABEL 6
HASIL TRAINING WORD2VEC CNN 2

Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	0,3532	0,8491	0,2421	0,9110
2	0,1251	0,9530	0,2176	0,9041
3	0,0308	0,9908	0,2522	0,9178
4	0,0090	0,9969	0,3161	0,8973
5	0,0043	0,9985	0,3781	0,8904

```
model_cnn_03 = Sequential()
e = Embedding(100000, 200, weights=[embedding_matrix], input_length=45, trainable=True)
model_cnn_03.add(e)
model_cnn_03.add(Conv1D(filters=100, kernel_size=2, padding='valid', activation='relu', strides=1))
model_cnn_03.add(GlobalMaxPooling1D())
model_cnn_03.add(Dense(256, activation='relu'))
model_cnn_03.add(Dense(1, activation='sigmoid'))
model_cnn_03.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model_cnn_03.fit(x_train_seq, y_train, validation_data=(x_val_seq, y_validation), epochs=5, batch_size=32, verbose=2)
```

Gbr. 30 Model training Word2vec CNN 3

TABEL 7
HASIL TRAINING WORD2VEC CNN 3

Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	0,4167	0,8056	0,3143	0,8699
2	0,2170	0,9161	0,2817	0,8973
3	0,0899	0,9702	0,2988	0,8836
4	0,0410	0,9860	0,2867	0,9041
5	0,0146	0,9958	0,3472	0,9110

Akurasi terbaik dari model Word2vec CNN adalah model Word2vec CNN 2 pada epoch 3 dengan akurasi validasi sebesar 91,78%.

Model selanjutnya menambahkan satu lapisan tersembunyi yang terhubung sepenuhnya dengan putus sebelum lapisan keluaran, dan juga lapisan keluaran hanya akan memiliki satu simpul keluaran, dengan aktivasi Sigmoid sebagai gantinya menggunakan API model sequential pada keras, dan diteruskan modelcheckpoint untuk mendapatkan nilai API model sequential keras yang dapat dilihat dibawah ini.

```
from keras.layers import Input, Dense, concatenate, Activation
from keras.models import Model

tweet_input = Input(shape=(45,), dtype='int32')

tweet_encoder = Embedding(100000, 200, weights=[embedding_matrix], input_length=45, trainable=True)(tweet_input)
bigram_branch = Conv1D(filters=100, kernel_size=2, padding='valid', activation='relu', strides=1)(tweet_encoder)
bigram_branch = GlobalMaxPooling1D()(bigram_branch)
trigram_branch = Conv1D(filters=100, kernel_size=3, padding='valid', activation='relu', strides=1)(tweet_encoder)
trigram_branch = GlobalMaxPooling1D()(trigram_branch)
fourgram_branch = Conv1D(filters=100, kernel_size=4, padding='valid', activation='relu', strides=1)(tweet_encoder)
fourgram_branch = GlobalMaxPooling1D()(fourgram_branch)
merged = concatenate([bigram_branch, trigram_branch, fourgram_branch], axis=1)

merged = Dense(256, activation='relu')(merged)
merged = Dropout(0.2)(merged)
merged = Dense(1)(merged)
output = Activation('sigmoid')(merged)
model = Model(inputs=[tweet_input], outputs=[output])
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])
model.summary()
```

Gbr. 31 Kode program model API keras

```
from keras.callbacks import ModelCheckpoint

filepath="CNN_best_weights.{epoch:02d}-{val_accuracy:.4f}.hdf5"
checkpoint = ModelCheckpoint(filepath, monitor='val_acc', verbose=1, mode='auto')

model.fit(x_train_seq, y_train, batch_size=32, epochs=5,
                    validation_data=(x_val_seq, y_validation), callbacks = [checkpoint])
```

Gbr. 32 Kode program modelcheckpoint

TABEL 8
HASIL MODEL CHECKPOINT

Epoch	Loss	Accuracy	Val_loss	Val_accuracy
1	0,4377	0,7969	0,3501	0,8630
2	0,2223	0,9136	0,2825	0,8767
3	0,0856	0,9699	0,2825	0,9110
4	0,0312	0,9884	0,4365	0,9178
5	0,0294	0,9907	0,3150	0,9247

Pada modelcheckpoint dapat dilihat untuk akurasi validasi mendapatkan nilai terbaik yaitu 92,47%, sedikit lebih baik dari model CNN sederhana dengan filter bigram yaitu 91,78%. untuk menampilkan nilai yang diinginkan maka digunakan fungsi seperti dibawah ini.

```
precision = precision_score(y_test, y_predict_cnn)
recall = recall_score(y_test, y_predict_cnn)
f1score = f1_score(y_test, y_predict_cnn)
accuracy = accuracy_score(y_test, y_predict_cnn)

print(f"Accuracy = {accuracy}")
print(f"Precision = {precision}")
print(f"Recall = {recall}")
print(f"F1 Score = {f1score}")
```

```
Accuracy = 0.8843537414965986
Precision = 0.9438202247191011
Recall = 0.875
F1 Score = 0.908108108108108
```

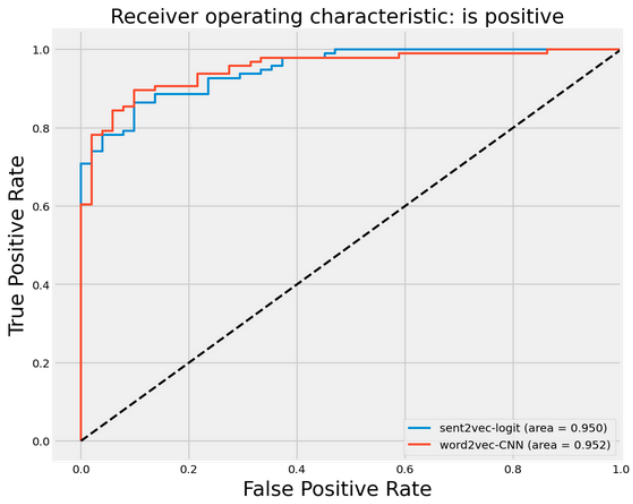
Gbr. 33 Kode program tampilan nilai

Selanjutnya mencari nilai kuvra ROC yang ada pada model Sent2vec LR dan Word2vec CNN yaitu dengan menggunakan fungsi dibawah ini.

```
from sklearn.metrics import roc_curve, auc
fpr, tpr, threshold = roc_curve(y_test, yhat_lr[:,1])
roc_auc = auc(fpr, tpr)
fpr_cnn, tpr_cnn, threshold = roc_curve(y_test, yhat_cnn)
roc_auc_nn = auc(fpr_cnn, tpr_cnn)
plt.figure(figsize=(8,7))
plt.plot(fpr, tpr, label='sent2vec-logit (area = %0.3f)' % roc_auc, linewidth=2)
plt.plot(fpr_cnn, tpr_cnn, label='word2vec-CNN (area = %0.3f)' % roc_auc_nn, linewidth=2)

plt.plot([0, 1], [0, 1], 'k--', linewidth=2)
plt.xlim([-0.05, 1.0])
plt.ylim([-0.05, 1.05])
plt.xlabel('False Positive Rate', fontsize=18)
plt.ylabel('True Positive Rate', fontsize=18)
plt.title('Receiver operating characteristic: is positive', fontsize=18)
plt.legend(loc="lower right")
plt.show()
```

Gbr. 34 Kode program mencari nilai kuvra ROC



Gbr. 35 Tampilan kuvra ROC

Pada kuvra ROC dapat dilihat untuk nilai Sent2vec LR mendapatkan nilai area 0,950 dan untuk Word2vec CNN mendapatkan nilai 0,952, dari perbedaan diatas maka dapat dilihat jika wilayah untuk kuvra lebih besar 0,002 dengan nilai tersebut maka Word2vec CNN memiliki area yang lebih besar dan menjadikan model tersebut lebih baik dibandingkan Sent2vec LR.

Setelah melakukan training kedua model maka dapat dilihat nilai untuk accuracy, precision, recall, F1 score, dan ROC pada tabel dibawah ini.

TABEL 9
HASIL PERBANDINGAN MODEL

Model	Accuracy	Precision	Recall	F1 Score	ROC
Sent2vec LR	84,35%	87,63%	88,54%	88%	95%
Word2vec CNN	88,44%	94,38%	88,6%	90,81%	95,2%

Pada tabel diatas dapat dilihat bahwa model Word2vec (CBOW) dengan metode CNN bisa mengungguli bentuk sederhana dari Sent2vec (TF-IDFF) + Logistic Regresion, dengan selisih hasil yaitu untuk Accuracy = 4,09%, Precision = 6,75%, Recall = 0,06%, F1-Score = 2,81%, ROC = 0,2%.

V. KESIMPULAN

Scrapping data Youtube Comment di salah satu video Channel Deddy Corbuzier yang berjudul "Nadiem, Kalau Bodo Satu Generasi Gimana Bro? – NADIEM MAKARIM – Deddy Corbuzier Podcast" dengan menggunakan *Google Spreadsheet* dan *Apps Script* mendapatkan hasil data total yaitu sebanyak 14.603 kalimat. Melakukan *Sentiment Analysis* untuk mendapatkan nilai positif dan negatif pada hasil *Scrapping Data* pada *YouTube Comment* di RStudio dengan melakukan tahapan Cleaning text yang tidak diperlukan dan menggunakan Stopwords yang didefinisikan sendiri, lalu melakukan penilaian dengan menggunakan data positif dan negatif yang

didefinisikan sendiri, mendapatkan data hasil labelling yaitu dengan total data sebanyak 9.549 kalimat positif dan 5.056 kalimat negatif. Melakukan perbandingan untuk mendapatkan hasil, yaitu tingkat *Accuracy, Prcison, Recall, F1 Score, Dan Roc* untuk *Model Word Embedding Sent2vec Logistic Regresion* dan *Word2vec Convolutional Neural Network* pada hasil *Sentiment Analysis* dengan menggunakan Jupyter Notebook. Setelah melakukan proses pada Jupyter Notebook untuk Perbandingan model didapatkan hasil yang memuaskan, yaitu untuk model Sent2vec (TF-IDFF) + *Logistic Regresion* mendapatkan hasil nilai Accuracy = 84,35%, Precission = 87,63%, Recall = 88,54%, F1 Score = 88%, ROC = 95%, sedangkan untuk model Word2vec (CBOW) dengan metode *CNN* mendapatkan hasil yang lebih baik yaitu nilai Accuracy = 88,44%, Precission = 94,38%, Recall = 88,6%, F1 Score = 90,81% dan nilai ROC = 95,2%.Untuk model Word2vec (CBOW) dengan metode *Cnn* menggunakan berbagai kombinasi program untuk memperoleh nilai. Maka diketahui bahwa model terbaik adalah model menggunakan *API Model* Pada *Keras* Dan Melalui *Modelcheckpoint* pada keras untuk mendapat hasil nilai dari model. Pada penelitian ini didapatkan model *Word Embedding* dengan terusan metode yang mempunyai nilai tinggi, sehingga sangat baik untuk dijadikan contoh model sebagai perbandingan model yang lain ataupun untuk melakukan Sentiment Analysis untuk *labelling* suatu data.

UCAPAN TERIMA KASIH

Setelah menyelesaikan penelitian ini penulis mengucapkan syukur kepada Allah ta'ala dan rosul tercinta Muhammad SAW, karena ini semua berkat usaha dan berkah-Nya, serta kepada kedua orang tua yang telah mensupport secara penuh kepada penulis, dan yang terakhir kepada dosen pembimbing dan seluruh dosen yang ada pada jurusan tercinta Teknik Informatika UNESA.

REFERENSI

[1] APJII, B. 2021. (2021). Tiga Syarat Jadi Hub Internet Dunia. *Apjii*, 1–10.

[2] Hendriyani, M., Artini, N. M., & Tatyana, T. (2021). Dampak Pandemi Covid 19 Terhadap Dunia Pendidikan. *Kompleksitas: Jurnal Ilmiah Manajemen, Organisasi Dan Bisnis*, *10*(2), 13–21. https://doi.org/10.56486/kompleksitas.vol10no2.128

[3] Indahri, Y. (2020). Permasalahan Pembelajaran Jarak Jauh di Era Pandemi. *Info Singkat: Kajian Singkat Terhadap Isu Aktual Dan Strategis*, *12*(2), 14–15. https://berkas.dpr.go.id/puslit/files/info_singkat/Info Singkat-XII-12-II-P3DI-Juni-2020-201.pdf

[4] Tanesab, F. I., Sembiring, I., & Purnomo, H. D. (2017). Sentiment Analysis Model Based On Youtube Comment Using Support Vector Machine. *International Journal of Computer Science and Software Engineering (IJCSSE)*, *6*(8), 180–185. http://ijcsse.org/published/volume6/issue8/p2-V6I8.pdf

[5] Baskara, R., & Rahma, F. (2022). Implementasi Web Scraping Pada Media Sosial Instagram. *Automata*, *3*, 1–3.

[6] Atikah, R., Prihatin, R. T., Hernayati, H., & Misbah, J. (2021). Pemanfaatan Google Classroom Sebagai Media. *Jurnal PETIK*, *7*(1), 7–18.

[7] Nababan, D. (2021). Sentimen Analisis Terhadap Kebijakan Pembelajaran Jarak Jauh Selama Pandemi Covid-19 Menggunakan Algoritma Naïve Bayes. *Jurnal Teknik Informatika*, *14*(1), 51–56. https://doi.org/10.15408/jti.v14i1.17002

[8] Ruger, A. H., Suyanto, M., & Kurniawan, M. P. (2021). Sentimen Analisis Pelanggan Shopee di Twitter dengan Algoritma Naive Bayes. *Journal of Information Technology*, *1*(2), 26–29. https://doi.org/10.46229/jifotech.v1i2.282

[9] Nurdin, A., Anggo Seno Aji, B., Bustamin, A., & Abidin, Z. (2020). Perbandingan Kinerja Word Embedding Word2Vec, Glove, Dan Fasttext Pada Klasifikasi Teks. *Jurnal Tekno Kompak*, *14*(2), 74. https://doi.org/10.33365/jtk.v14i2.732

[10] Prameswari, K., & Setiawan, E. B. (2019). Analisis Kepribadian Melalui Twitter Menggunakan Metode Logistic Regression dengan Pembobotan TF-IDF dan AHP. *E-Proceeding of Engineering*, *6*(2), 9667–9682.

[11] Panyahuti, P., & Yadi, Y. (2022). Pengembangan Aplikasi E-Assessment Skill Programming berbasis Web. *Edumatic: Jurnal Pendidikan Informatika*, *6*(1), 78–87. https://doi.org/10.29408/edumatic.v6i1.5393

[12] Anindyati, L., Purwarianti, A., & Nursanti, A. (2019). Optimizing Deep Learning for Detection Cyberbullying Text in Indonesian Language. *Proceedings - 2019 International Conference on Advanced Informatics: Concepts, Theory, and Applications, ICAICTA 2019*, 1–5. https://doi.org/10.1109/ICAICTA.2019.8904108

[13] Kiran, R., Kumar, P., & Bhasker, B. (2020). Oslcfit (organic simultaneous LSTM and CNN Fit): A novel deep learning based solution for sentiment polarity classification of reviews. *Expert Systems with Applications*, *157*, 113488. https://doi.org/10.1016/j.eswa.2020.113488

[14] Maulana, A. R., & Rochmawati, N. (2020). Opinion Mining Terhadap Pemberitaan Corona di Instagram menggunakan Convolutional Neural Network. *Journal of Informatics and Computer Science (JINACS)*, *2*(01), 53–59. https://doi.org/10.26740/jinacs.v2n01.p53-59

[15] Muhammad Arief Rahman, Herman Budianto, & Esther Irawati Setiawan. (2019). Aspect Based Sentimen Analysis Opini Publik Pada Instagram dengan Convolutional Neural Network. *Journal of Intelligent System and Computation*, *1*(2), 50–57. https://doi.org/10.52985/insyst.v1i2.83

[16] Nasichuddin, M. A., Adji, T. B., & Widyawan, W. (2018). Performance Improvement Using CNN for Sentiment Analysis. *IJITEE (International Journal of Information Technology and Electrical Engineering)*, *2*(1). https://doi.org/10.22146/ijitee.36642

[17] S., S., & K.V., P. (2020). Sentiment analysis of malayalam tweets using machine learning techniques. *ICT Express*, *6*(4), 300–305. https://doi.org/10.1016/j.icte.2020.04.003

[18] S., S., & K.V., P. (2020). Sentiment analysis of malayalam tweets using machine learning techniques. *ICT Express*, *6*(4), 300–305. https://doi.org/10.1016/j.icte.2020.04.003

[19] Yue, W., & Li, L. (2020). Sentiment Analysis using Word2vec-CNN-BiLSTM Classification. 3–7.