BAB IV
IMPLEMENTASI DAN PENGUJIAN SISTEM

4.1 Implementasi

Tahap implementasi merupakan suatu penerapan dari setiap proses atau rancangan siste, yang telah dibuat sebelumnya sehingga menghasilkan sistem yang sesuai dengan kebutuhan yang sudah diatur sebelumnya

4.1.1 Implementasi Perangkat Pendukung

Untuk melakukan implementasi dan pengujian sistem aplikasi dibutuhkan perangkat keras (*hardware*) dan perangkat lunak (*software*). Berikut adalah perangkat keras (*hardware*) dan perangkat lunak (*software*) yang digunakan dalam melakukan implementasi dan pengujian pada aplikasi *chatbot* Jasa Rental Alat Kamping.

1. Perangkat Keras (*Hardware*)

   Berikut adalah spesifikasi Perangkat Keras (*Hardware*) yang akan digunakan :

   a. RAM 4 GB
   b. *Harddisk* 500 GB
   c. *Processor AMD A6 6400K 3.50 Ghz*
   d. *Keyboard* dan *Mouse*

2. Perangkat Lunak (*Software*)

   Berikut adalah spesifikasi Perangkat Lunak (Software) yang akan digunakan :

   a. Sistem Operasi *Windows 10*
   b. Bahasa Pemrograman *Python*
   c. *Visual Studio Code* dan *Sublime text*
   d. *Power Designer*
   e. *Google Chrome Browser*

4.1.2 Implementasi *Preprocessing*

Proses awal dalam pengkodean adalah dengan persiapan data dan membersihkan data. Data yang telah dikumpulkan dan diberi kelas pada setiap pertanyaanya akan diproses untuk ke tahap selanjutnya.

1. Membaca Dataset

   Membaca dataset rental alat kamping dari file xltx dan menyimpan kedalam dataframe.

```
import pandas as pd
data=pd.read_excel('/content/drive/MyDrive/cnn/dataset_super_final.xltx')data
```

Dataset dari file xltx kemudian disimpan dalam variable *data*. Variabel *data* dimasukan kedalam kerangka dataframe menggunakan *library pandas*.

	label	question	answer
0	carier	Maaf kang carier itu apa?	Carier adalah suatu alat atau tas dengan kapas...
1	carier	Apa daypack itu?	Daypack adalah tempat atau alat yang digunakan...
2	carier	Daypack yang ada kapasitas berapa saja?	Kapasitas daypack hanya 30liter saja.
3	carier	Kang carier yang tersedia tuh kapasitas berapa...	Untuk kapasitas carier yang tersedia dimulai d...
4	carier	Rekomendasi dong kang carier buat perjalanan 1...	Rekomendasi carier untuk perjalan 1-2 adalah c...
...	...	...	...
195	persyaratan	Bisa ga kang pesyaratan yang diserahkan BPKB?	Untuk BPKB belum bisa diserahkan sebagai persy...
196	persyaratan	Bisa ga kang pesyaratan yang diserahkan sertif...	Untuk Sertifikat belum bisa diserahkan sebagai...
197	persyaratan	Bisa ga kang pesyaratan yang diserahkan barang...	Untuk Barang antik belum bisa diserahkan sebag...
198	persyaratan	Bisa ga kang pesyaratan yang diserahkan handph...	Untuk Handphone bisa diserahkan sebagai persya...
199	persyaratan	Bisa ga kang pesyaratan yang diserahkan laptop?	Untuk Laptop bisa diserahkan sebagai persyarat...

200 rows × 3 columns

Gambar 4.1 Hasil Baca Dataset

Pada Gambar 4.1 Adalah hasil dari proses baca dataset dan memasukan kedalam dataframe. Dataset rental alat kamping dibagi menjadi 3 kolom yaitu : *label* (kelas dari pertanyaan), *question* (pertanyaan tentang rental alat kamping), dan *answers* (jawaban dari satu baris pertanyaan).

2. *Cleaning* dan *Case Folding*

Membersihkan dataset dari tanda baca dan merubah kalimat pada kolom pertanyaan menjadi huruf kecil.

```
def preprocess_text(text):
cleaned_text = re.sub(r'[^\w\s]', '', text)
folded_text = cleaned_text.lower
return folded_text
```

Gambar 4.2 Merupakan hasil dari proses *cleaning* dan *case folding*.

```
Hasil Cleaning dan Case-Folding
                                                 question
0                              Maaf kang carier itu apa?
1                                    Apa daypack itu?
2                Daypack yang ada kapasitas berapa saja?
3      Kang carier yang tersedia tuh kapasitas berapa...
4      Rekomendasi dong kang carier buat perjalanan 1...
..                                                   ...
195        Bisa ga kang pesyaratan yang diserahkan BPKB?
196    Bisa ga kang pesyaratan yang diserahkan sertif...
197    Bisa ga kang pesyaratan yang diserahkan barang...
198    Bisa ga kang pesyaratan yang diserahkan handph...
199      Bisa ga kang pesyaratan yang diserahkan laptop?

                                          cleaned_question
0                              maaf kang carier itu apa
1                                    apa daypack itu
2                daypack yang ada kapasitas berapa saja
3      kang carier yang tersedia tuh kapasitas berapa...
4      rekomendasi dong kang carier buat perjalanan 1...
..                                                   ...
195        bisa ga kang pesyaratan yang diserahkan bpkb
196    bisa ga kang pesyaratan yang diserahkan sertif...
197    bisa ga kang pesyaratan yang diserahkan barang...
198    bisa ga kang pesyaratan yang diserahkan handphone
199      bisa ga kang pesyaratan yang diserahkan laptop

[200 rows x 2 columns]
```

Gambar 4.2 *Cleaning* dan *Case Folding*.

3. *Tokenizing*

Proses selanjutnya adalah dengan memisahkan kata pada setiap kalimat.

```
def tokenize_text(text):
tokens = word_tokenize(text)
return tokens
```

Untuk melakukan tokenizing dapat menggunakan *library nltk*. Fungsi *word_tokenize* merubah kalimat menjadi susunan kata yang saling terpisah. Hasil *tokenizing* disimpan dalam *df*[‘*tokenize’*]. Gambar 4. Merupakan hasil dari *tokenizing*.

```
Hasil Tokenizing
                                      cleaned_question  \
0                              maaf kang carier itu apa
1                                       apa daypack itu
2                daypack yang ada kapasitas berapa saja
3     kang carier yang tersedia tuh kapasitas berapa...
4     rekomendasi dong kang carier buat perjalanan 1...
..                                                  ...
195       bisa ga kang pesyaratan yang diserahkan bpkb
196  bisa ga kang pesyaratan yang diserahkan sertif...
197  bisa ga kang pesyaratan yang diserahkan barang...
198  bisa ga kang pesyaratan yang diserahkan handphone
199     bisa ga kang pesyaratan yang diserahkan laptop

                                     tokenized_question
0                        [maaf, kang, carier, itu, apa]
1                                  [apa, daypack, itu]
2          [daypack, yang, ada, kapasitas, berapa, saja]
3     [kang, carier, yang, tersedia, tuh, kapasitas,...
4     [rekomendasi, dong, kang, carier, buat, perjal...
..                                                  ...
195  [bisa, ga, kang, pesyaratan, yang, diserahkan,...
196  [bisa, ga, kang, pesyaratan, yang, diserahkan,...
197  [bisa, ga, kang, pesyaratan, yang, diserahkan,...
198  [bisa, ga, kang, pesyaratan, yang, diserahkan,...
199  [bisa, ga, kang, pesyaratan, yang, diserahkan,...

[200 rows x 2 columns]
```

Gambar 4.3 Hasil *Tokenizing*

4. *Spelling Normalization*

   Proses ini dilakukan untuk mengganti kata yang salah eja atau singkatan singkatan pada kata.

   from nltk.tokenize.treebank import TreebankWordDetokenizer # Fungsi untuk melakukan normalisasi def normalized_term(document): return [normalizad_word_dict[term] if term in normalizad_word_dict else term for term in document]

```
text_detokenize =
TreebankWordDetokenizer().detokenize(text)
return text_detokenize
normalizad_word_dict = {}
for index, row in normalizad_word.iterrows():
if row[0] not in normalizad_word_dict:
normalizad_word_dict[row[0]] = row[1]
df['normalized'] = df['tokenize'].apply(normalized_term)
df['normalized'] = df['normalized'].apply(lambda x:
detoken(x))
```

   Berikut adalah hasil dari *spelling normalization*.

	label	question	answer
0	carier	maaf kang carier itu apa	carier adalah suatu alat atau tas dengan kapas...
1	carier	apa daypack itu	daypack adalah tempat atau alat yang digunakan...
2	carier	daypack yang ada kapasitas berapa saja	kapasitas daypack hanya 30 liter saja.
3	carier	kang carier yang tersedia tuh kapasitas berapa...	untuk kapasitas carier yang tersedia dimulai d...
4	carier	rekomendasi kang carier buat perjalanan 12hari	rekomendasi carier untuk perjalan 12 adalah ca...
...	...	...	...
195	persyaratan	bisa tidak kang pesyaratan yang diserahkan bpkb	untuk bpkb belum bisa diserahkan sebagai persy...
196	persyaratan	bisa tidak kang pesyaratan yang diserahkan ser...	untuk sertifikat belum bisa diserahkan sebagai...
197	persyaratan	bisa tidak kang pesyaratan yang diserahkan bar...	untuk barang antik belum bisa diserahkan sebag...
198	persyaratan	bisa tidak kang pesyaratan yang diserahkan han...	untuk handphone bisa diserahkan sebagai persya...
199	persyaratan	bisa tidak kang pesyaratan yang diserahkan laptop	untuk laptop bisa diserahkan sebagai persyarat...

200 rows × 3 columns

Gambar 4.4 Hasil *Spelling Normalization*

4.1.3 Implementasi Model

Model ini dibuat untuk mengklasifikasikan intent pada dataset rental kamping. Berikut tahapan pembuatan model menggunakan algoritma CNN.

1. *Word Embedding* menggunakan *FastText*

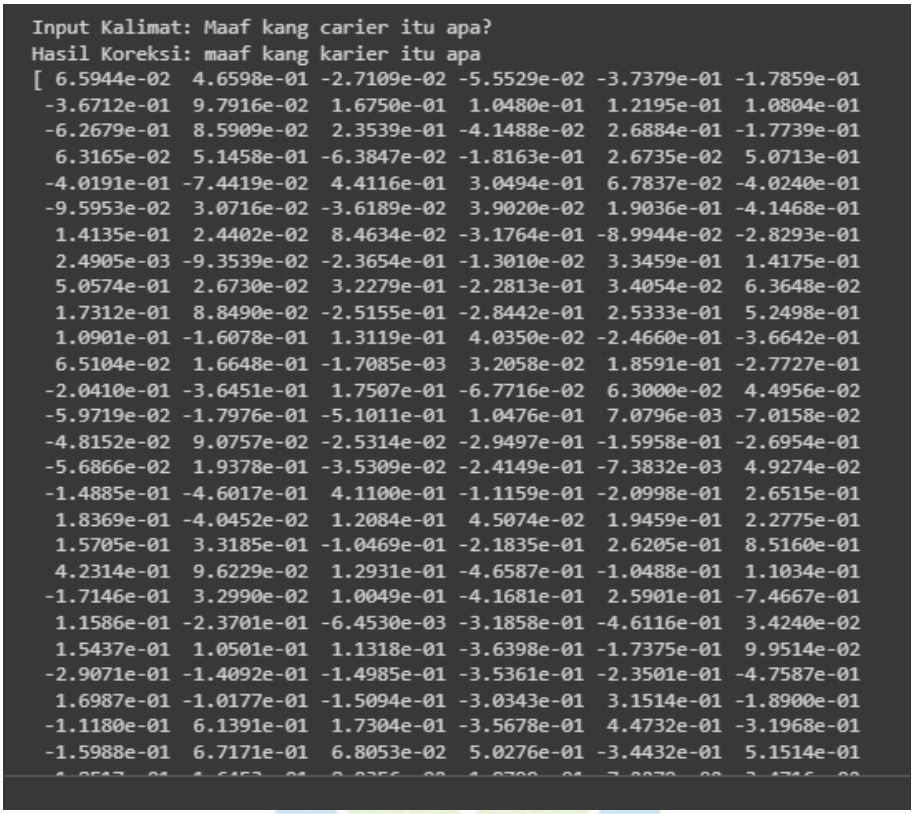
   Word embedding digunakan untuk merepresentasikan kata ke dalam vektor padat. Hasil training korpus menggunakan *FastText* di load kemudian di masukan kedalam *embedding* layer. Berikut adalah implementasi potongan *source Word Embedding* menggunakan *FastText*.

```
file_vektor_dari_model =
'/content/drive/MyDrive/cnn/wiki.id.vec'
vektor_dari_kata =
KeyedVectors.load_word2vec_format(file_vektor_dari_model,
limit=0)

waktu_selesai = time.time()
waktu_proses = waktu_selesai - waktu_mulai
print("Waktu proses: ", waktu_proses)

# komputasi untuk mendapatkan vektor dari setiap kata
for kata in hasil_koreksi.split():
    if kata in vektor_dari_kata:
word_embedding_kata = vektor_dari_kata[kata]
print(word_embedding_kata)
```


Hasil dari *word embedding* menggunakan *fasttext* dapat dilihat pada gambar 4.4



Gambar 4.5 Hasil *Word Embedding* menggunakan *FastText*

2. Implementasi Algoritma CNN

Model *Convolutional Neural Network* pada penelitian ini dimodelkan dengan memiliki satu lapis *layer* konvolusi 1D, satu lapis *global max pooling layer*, dan dua *layer dense*. Berikut adalah potongan *source* implementasi CNN.

```
def build cnn_model():
model = Sequential()

#Proses Embedding layer
model.add(Embedding(vocab_size, EMBED_SIZE,
input_length=MAX_SEQUENCE_LENGTH))

#Proses Konvolusi
model.add(Conv1D(filters=128, kernel_size=3,
padding='same', activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling1D(pool_size=2))
```

```
model.add(Conv1D(filters=64, kernel_size=3,
padding='same', activation='relu'))
model.add(BatchNormalization())

#Proses Pooling
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=32, kernel_size=3,
padding='same', activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())

#Proses Fully Conected
model.add(Dense(256, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

Berikut adalah hasil compile model CNN dapat dilihat pada gambar 4.6

```
Model: "sequential_39"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_37 (Conv1D)          (None, 30, 100)           90100

 global_average_pooling1d_23  (None, 100)              0
  (GlobalAveragePooling1D)

 dense_70 (Dense)            (None, 20)                2020

 dense_71 (Dense)            (None, 1)                 21

=================================================================
Total params: 92,141
Trainable params: 92,141
Non-trainable params: 0
```

Gambar 4.6 compile model CNN

3. Implementasi Klasifikasi *Intent*/kelas

Untuk mengklasifikasi *intent* dari pertanyaan yang diajukan *user* kedalam 6 kelas yang sudah ditentukan, maka model yang telah dibuat bisa diimplementasikan. Berikut adalah implementasi potongan *source* prediksi *intent*/kelas.

```
model.predict(pad_sequences(tokenizer.texts_to_sequences([
hasil_koreksi]),truncating='post', maxlen=max_len))
tag = encoder.inverse_transform([np.argmax(result)])
x = re.search(r"\w+", str(tag))
```

4.1.4 Implementasi *Cosine Similarity*

Cosine similarity digunakan untuk menghitung nilai kesamaan *keyword* pertanyaan dengan *keyword* yang ada pada dataset. Berikut adalah implementasi potongan *source cosine similarity*.

```
# pilih jawaban berdasarkan kelas
response = data[data['label'] == x.group()]

# skor pada setiap pertanyaan, pertanyaan dg skor tertinggi
(semakin besar skornya maka semakin mirip dg pertanyaan yg
diajukan user)
response['skor'] = response.apply(lambda row:
SequenceMatcher(None, hasil_koreksi, row['question']).ratio(),
axis = 1)

# pilih skor terbesar (semakin skor yg besar artinya pertanyaan
yg diajukan user hampir mirip dengan pertanyaan di dataset)
tes = response.loc[response['skor'].idxmax()]
print("jawaban: ", tes.answer)
```


4.2 Pengujian Model

4.2.1 Pengujian Model CNN

Pengujian algoritma dilakukan untuk mengetahui akurasi dengan menggunakan algoritma CNN dalam mengklasifikasi *intent* pada 6 kelas yang sudah diatur. Pengujian ini melibatkan parameter pembagian dataset dan jumlah *epoch* yang digunakan dalam proses pelatihan dan evaluasi algoritma. Tujuan dari pengujian ini adalah untuk mengetahui seberapa baik algoritma CNN dalam melakukan klasifikasi intent pada dataset yang telah ditentukan..

Epoch adalah banyaknya jumlah pengulangan saat proses pelatihan model. Jumlah *epoch* yang berbeda bertujuan untuk memperoleh hasil yang optimal dan jika *epoch* terlalu banyak maka akan *overlooping*. Pada pengujian ini dilakukan sebanyak 3 kali pengujian dengan memakai 90% data latih, 10% data uji, dan *epoch*

10, 90% data latih, 10% data uji, dan *epoch* 20, kemudian 90% data latih, 10% data uji, dan *epoch* 50.

1. Pengujian Pertama

Pengujian pertama dilakukan dengan membagi dataset menjadi 90% data latih sebanyak 180 data dan 10% data uji sebanyak 20 data menggunakan *epoch* 10.

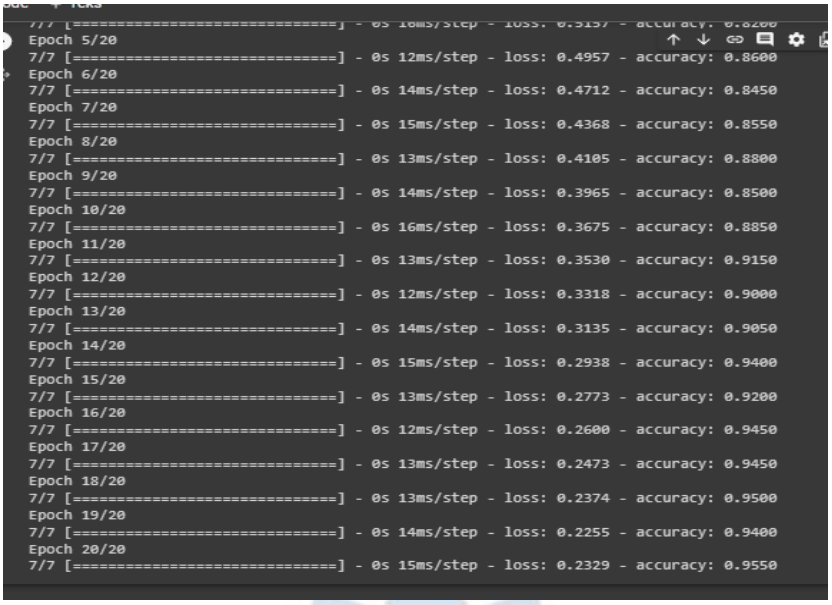
```
epochs = 10
history = model.fit(padded_sequences, np.array(training_labels), epochs=epochs)

Epoch 1/10
7/7 [==============================] - 1s 12ms/step - loss: 1.7792 - accuracy: 0.2450
Epoch 2/10
7/7 [==============================] - 0s 14ms/step - loss: 1.7277 - accuracy: 0.3150
Epoch 3/10
7/7 [==============================] - 0s 13ms/step - loss: 1.6630 - accuracy: 0.4400
Epoch 4/10
7/7 [==============================] - 0s 18ms/step - loss: 1.5682 - accuracy: 0.5650
Epoch 5/10
7/7 [==============================] - 0s 13ms/step - loss: 1.4425 - accuracy: 0.6200
Epoch 6/10
7/7 [==============================] - 0s 14ms/step - loss: 1.2927 - accuracy: 0.6800
Epoch 7/10
7/7 [==============================] - 0s 12ms/step - loss: 1.1238 - accuracy: 0.7550
Epoch 8/10
7/7 [==============================] - 0s 14ms/step - loss: 0.9439 - accuracy: 0.7750
Epoch 9/10
7/7 [==============================] - 0s 12ms/step - loss: 0.8081 - accuracy: 0.7900
Epoch 10/10
7/7 [==============================] - 0s 12ms/step - loss: 0.7148 - accuracy: 0.7950
```

Gambar 4.7 Hasil Pengujian Pertama

2. Pengujian Kedua

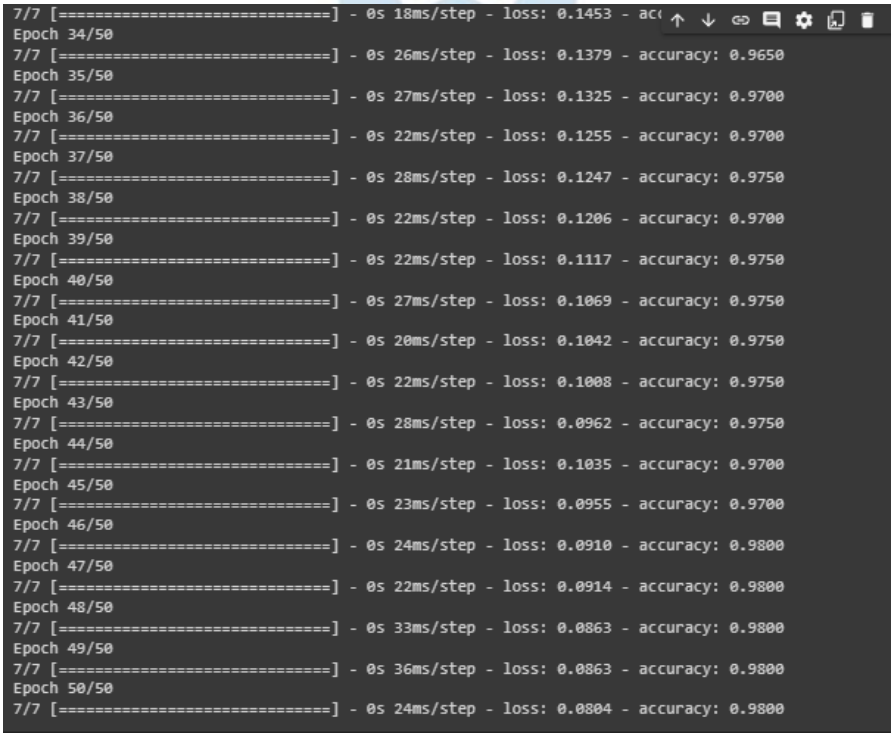
Pengujian kedua dilakukan dengan membagi dataset menjadi 90% data latih sebanyak 180 data dan 10% data uji sebanyak 20 data menggunakan *epoch* 20.



Gambar 4.8 Hasil Pengujian Kedua

3. Pengujian Ketiga

Pengujian ketiga dilakukan dengan membagi dataset menjadi 90% data latih sebanyak 180 data dan 10% data uji sebanyak 20 data menggunakan *epoch* 50.



Gambar 4.9 Hasil Pengujian Ketiga

4.3 Evaluasi Sistem

Aplikasi *chatbot* sudah sesuai dengan rancangan *prototype* dan bisa digunakan. Algoritma CNN sudah diimplementasikan pada sistem dan digunakan untuk mengkalsifikasi *intent* kedalam kelas yang sudah diatur. Begitu pula dengan algoritma tambahan seperti *cosine similarity* sudah diimplementasikan dengan baik pada aplikasi *chatbot* rental alat kamping.

Tabel 4.1 Hasil Pengujian CNN

Data Latih		**Data Uji**		**Akurasi (%)**		
				Epoch		
%	**Data**	**%**	**Data**	**10**	**20**	**50**
90	180	10	20	79.5%	95.5%	98.0%

Setelah melalui 3 kali pengujian model CNN, dapat diketahui bahwa akurasi tertinggi diperoleh dengan pembagian dataset menjadi 90% data latih dan 10% data uji dengan menggunakan 50 *epoch*. Sedangkan hasil akurasi terendah didapatkan dengan pembagian dataset 90% data latih dan 10% data uji dengan menggunakan *epoch* 10.

Pada pengujian ini memiliki 200 dataset, sangat sedikit untuk sebuah pemodelan CNN. Akurasi tidak dapat mencapai 100% dikarenakan banyak faktor, salah satunya adalah keterbatasan jumlah dataset yang terbilang sangat sedikit serta ketidak seimbangan jumlah data pada setiap kelas.