JIKO (Jurnal Informatika dan Komputer) Akreditasi KEMENRISTEKDIKTI, No.36/E/KPT/2019
Vol. 5, No. 1, April 2022, hlm. 19-27 p-ISSN: 2614-8897
DOI:10.33387/jiko e-ISSN: 2656-1948

APLIKASI *CHATBOT* UNTUK FAQ AKADEMIK DI IBI-K57 DENGAN LSTM DAN PENYEMATAN KATA

Astried Silvanie[1], Rino Subekti[2]

[1,2]Program Studi Informatika, Fakultas Ilmu Komputer, Institut Bisnis & Informatika (IBI) Kosgoro 1957
Email: [1]astried@ibi-k57.ac.id , [2]rino.subekti@ibik57.ac.id

(Naskah masuk: 5 November 2021, diterima untuk diterbitkan: 23 November 2021)

Abstrak

Dalam kegiatan perkuliahan khususnya untuk mendapatkan informasi akademik, teknologi komputer saat ini sudah bisa memfasilitasi hal ini untuk dilakukan secara online. Tetapi jika ada pertanyaan dari mahasiswa maka peran karyawan masih diperlukan sebagai dukungan kustomer. Jika dilihat dari kemajuan dalam teknologi kecerdasan buatan khususnya pembelajaran mesin sangat dimungkinkan untuk membuat aplikasi cerdas yang dapat menjawab pertanyaan-pertanyaan secara otomatis tanpa bantuan manusia. Penelitian ini bertujuan membuat, melatih dan mengimplementasikan aplikasi FAQ chatterbot cerdas yang bisa memahami maksud dari kalimat kemudian menjawabnya. Algoritma yang digunakan adalah jaringan syaraf tiruan Long Short Term Memory (LSTM) dengan bantuan penyisipan kata menggunakan vektor kata yang sudah terlatih dalam model Continuous Bag Of Word (CBOW). Model JST dilatih dengan jumlah epochs 10, 20, 30, 40, 50, 60, 70, 80, 90 dan 100. Model dikompilasi dengan tiga jenis pengoptimal Adaptive Moment (adam), Root Mean Square Propagation (RMSprop) dan Stochastic Gradient Descent (SGD). Fungsi kerugian menggunakan categorical crossentropy dan metrik validasi akurasi. Nilai akurasi tertinggi berdasarkan pengoptimalnya adalah 99.20% untuk adam dan RMSprop sedangkan dengan pengoptimal SGD hanya sampai 12.90%. Ini membuktikan untuk model ini pengoptimal adam dan RMSprop lebih baik digunakan daripada dengan pengoptimal SGD. Implementasi dibangun dalam arsitektur client-server. Di sisi client dibangun aplikasi telepon gengam berbasis Android dengan bahasa pemrograman kotlin. Aplikasi ini digunakan pengguna akhir untuk bertanya mengenai masalah akademik. Di sisi server dibangun aplikasi dengan kerangka web Flask sebagai aplikasi python yang menjalankan input-proses-ouput menggunakan model JST. Aplikasi ini juga berfungsi memilih pasangan jawaban yang tepat dan kemudian mengirimkannya ke aplikasi pengguna akhir.

Kata kunci: *chatbot, LSTM, jaringan syaraf tiruan, Flask*

CHATBOT APPLICATION FOR ACADEMIC FAQ IN IBI-K57 WITH LSTM AND WORD EMBEDDING

Abstract

In lecture activities, especially to get academic information, the current computer technology has been able to facilitate this to be done online. But if there are questions from students, the role of employees is still needed as a customer support. When we see the progress in artificial intelligence technology, especially in machine learning, it is possible to create intelligent applications that can answer questions automatically without human assistance. This study aims to create, train and implement an intelligent chatterbot FAQ application that can understand the meaning of a sentence and then answer it. The algorithm used is a Long Short Term Memory (LSTM) artificial neural network with the help of word embedding using word vectors that have been trained as the Continuous Bag Of Word (CBOW) model. The ANN Model has been trained with epochs of 10, 20, 30, 40, 50, 60, 70, 80, 90 and 100. The model is compiled with three types of optimizer Adaptive Moment (adam), Root Mean Square Propagation (RMSprop) and Stochastic Gradient Descent (SGD). The loss function uses crossentropy categorization and accuracy assessment. The highest accuracy value based on the optimizer is 99.20% for adam and RMSprop, while the SGD optimizer is only able to achieve 12.90% accuracy. This proves for adam and RMSprop optimizer model are better to use than SGD optimizer. Implementation is built in a client-server architecture. On the client side, an Android-based mobile phone application is built using the Kotlin programming language. This application is used by end users to ask questions about academic problems. On the server side, the application is built with the Flask web framework as a python application that runs input-

process-output using the ANN model. This application also works on selecting the correct answer pairs and then sending them to the end user application.

Keywords: *chatbot, LSTM, artifical neural network, Flask*

1. PENDAHULUAN

Infrastruktur teknologi informasi dan komunikasi (TIK) sudah memungkinkan universitas menyediakan sistem secara online sebagai pendukung kegiatan perkuliahan dan beberapa aktivitas akademiknya [1]. Apalagi di masa pandemi ini peran TIK sangat membantu dalam kegiatan belajar mengajar dan pemrosesan informasi akademik. Tetapi sistem online ini belum tentu mempunyai fungsi *Frequently Asked Question* (FAQ) yang dapat menjawab secara otomatis. Jika ada pertanyaan baik dari mahasiswa atau dosen masih dilakukan oleh karyawan yang berperan sebagai pendukung kustomer dengan waktu kerja terbatas. Sistem FAQ secara otomatis ini dapat saja dibuat dengan bantuan komputer. Kelebihannya adalah komputer tidak mengenal lelah dan batasan waktu, sehingga bisa diakses kapan saja dan dimana saja [1]. Pertanyaannya bagaimana membuat sistem *Frequently Asked Question* (FAQ) yang dapat memahami maksud pertanyaan dan menjawabnya secara otomatis tanpa perlu keberadaan manusia. *Chatbot*, atau dikenal juga dengan *chatter robot,*adalah agen perangkat lunak yang dapat mensimulasikan percakapan manusia melalui teks atau pesan suara [2]. Seperti layaknya asisten virtual, *chatbot* membantu menjawab beberapa pertanyaan pengguna dan kemudian merespon sesuai dengan apa maksud dari pertanyaan tersebut. Sebuah FAQ *chatbot* adalah implementasi yang membandingkan pola di kalimat, dimana urutan kalimat tersebut dikenali dan kemudian merespon dengan pola respon yang tersimpan[3].

Teknologi chatbot ini termasuk kedalam kategori Natural Language Processing (NLP). Penelitian-penelitian khusus berkaitan dengan klasifikasi bahasa dengan NLP sudah cukup banyak. Pada [2] membuat chatbot sebagai asisten kesehatan virtual. Penggunaan Natural Language Processing (NLP) dan Deep Learning untuk membuat chatbot yang cerdas [4]. Penelitian [5] mengidentifikasikan sarkasme di media sosial menggunakan jaringan saraf tiruan dan penyisipan kata. Penelitian [6] membandingkan metode Naive Bayes dan Support Vector Machine untuk klasifikasi bahasa indonesia, jawa dan ambon. Penelitian-penelitian sebelum yang berfokus pada chatbot sebagai pelayan pelanggan adalah penelitian [3][7] tetapi kedua implementasi ini dilakukan untuk bahasa asing. Penelitian [8] membuat sebuah chatbot berbahasa indonesia untuk layanan akademik dengan metode K-Nearest Neighbor dan hasil akurasi hanya 53.48%. Selain itu penelitian [8] mengalami kendala adanya pertanyaan-pertanyaan dari kemiripan susunan kata dari kelas berbeda dan susahnya mengenali kata baku. Hal ini sebenarnya dapat dibantu dengan penggunaan penyematan kata. Penyematan kata atau dalam bahasa inggris Word Embedding dapat mengenali hubungan kata secara sintaksis dan semantik dari korpus yang sangat besar. Pada penelitian [9] dibuat chatbot untuk pelayan pelanggan hotel. Penelitian mereka berfokus pada pembuatan chatbot dengan Google Flutter sebagain front-end dan Artificial Intelligence Markup Language (AIML) sebagai back-end. Pada penelitian [10] [11] LSTM dikatakan baik untuk mendeteksi data yang tersusun berurutan dan model ini dapat memahami arti dalam kalimat dan akan memberikan kelas luaran yang paling akurat.

Penelitian ini membangun aplikasi Chatbot untuk menjawab pertanyaan-pertanyaan umum akademik yang sering ditanyakan untuk mahasiswa di Institut Bisnis dan Informatika Kosgoro 1957 (IBI-K57). Dalam penelitian, model jaringan syaraf tiruan LSTM dilatih sendiri menggunakan python dan jupyter notebook. LSTM dipilih karena untuk klasifikasi maksud model ini cukup akurat jika dibandingkan dengan RNN dan K-Nearest Neighbor. Kemudian kami menambahkan penyematan kata agar mampu mengenali kata yang secara sintaks dan semantik mirip. Setelah kami mendapatkan model JST yang sudah terlatih, model ini diimplementasikan dalam arsitektur client server dan bisa digunakan dengan aplikasi front end sebagai aplikasi yang digunakan pengguna akhir yaitu mahasiswa.

2. METODE PENELITIAN

2.1. Identifikasi Masalah

Dalam penelitian ini masalah yang ingin dijawab adalah (1) bagaimana membuat aplikasi *FAQ chatbot* cerdas untuk menjawab pertanyaan-pertanyaan akademik umum menggunakan algoritme pembelajaran mesin, dan (2) bagaimana mengimplementasikan *FAQ chatbot* tersebut dalam aplikasi telepon genggam dengan sistem operasi *Android*.

2.2. Klasifikasi Maksud

Jika diketahui kalimat s Є X dari sebuah kalimat dialog, di mana X adalah ruang ucapan dan K adalah jumlah dari kalimat s. Maka dapat dimodelkan seperti dibawah ini.

$$S = \{S_i \mid 1 \leq i \leq K\} \quad (1)$$

Satu set*Intent Class*sebanyak j didefinisikan sebagai berikut:

$$C = \{C_i \mid 1 \leq i \leq J\} \quad (2)$$

Satu set pelatihan S dengan label maksud:

$$\{s_i , c_i\}^{N}_{i=1} \text{ , di mana } (s, c) \text{ Є } X \times C \quad (3)$$

Setiap kalimat s akan diberi label tunggal atau dengan kata lain klasifikasi label tunggal, di mana setiap sberpasangan dengan satu dan hanya satu elemen dari C.

2.3. *Word Embedding*

Word Embedding atau penyematan kata adalah representasi vektor bernilai bilangan *real* untuk kosa kata berukuran tetap yang sebelumnya telah ditentukan dari kumpulan dari korpus besar yang tidak berlabel [12]. Penyematan kata digunakan dalam pemrosesan bahasa alami modern, seperti dalam analisis semantik, pencarian informasi, penguraian ketergantungan, penjawab pertanyaan, dan terjemahan mesin [13]. Salah satu model penyematan kata yaitu *Continuous Bag Of Word* (CBOW). Model CBOW mencoba memahami konteks dari kata-kata dan mengambil konteks ini sebagai masukan. Kemudian model ini akan mencoba untuk memprediksi apakah kata-kata ini secara kontekstual akurat [14].

Penelitian ini mengunakan vektor kata yang sudah dilatih oleh Kyubyong pada model *Word2vec* dari sumber teks Wikipedia. Vektor kata ini diunduh secara publik dari github Kyubyong (url: https://github.com/Kyubyong/wordvectors) dengan nama file id.tsv. Vektor ini terdiri atas 30.048 kata dalam bahasa indonesia. Setiap kata direpresentasikan sebagai vektor berukuran 1 x 300. Sebagai contoh diambil kata pertama yaitu *"yang"* direpresentasikan sebagai vektor berikut ini:

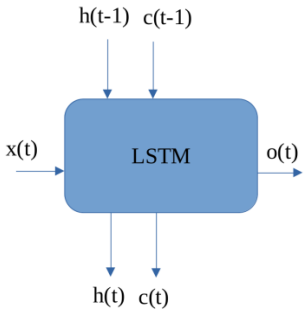
$$\text{yang} = [X_0, \ldots , X_{299}] \quad (4)$$

$$\text{yang} = [-0.074976, \ldots , 0.0068165] \quad (5)$$

2.4. *Long Short-Term Memory* (LTSM)

Dalam jaringan syaraf tiruan tradisional, semua input dan output bersifat independen satu sama lain, hal ini kurang tepat saat bekerja dengan informasi yang tersusun berurutan. Memprediksi kata berikutnya dalam sebuah kalimat membutuhkan pengetahuan urutan kata-kata dari kalimat sebelumnya. Salah satu model dari jaringan saraf *Recurrent Neural Network* (RNN), yaitu *Long Short Term Memory* (LSTM), adalah model yang paling tepat untuk memproses kalimat berurutan, dan biasa digunakan untuk proses kategorisasi teks dan terjemahan mesin [15]. *Long Short-Term Memory* (LSTM) adalah jaringan syaraf tiruan *Recurrent Neural Network* (RNN) di mana adanya gerbang yang digunakan untuk menghilangkan masalah gradien pada RNN tradisional [5]. LSTM menggunakan data historis dan data saat ini bersamaan untuk membuat prediksi. Hal ini dapat dicapai dengan menggunakan sel memori, di mana semua informasi masa lalu yang relevan disimpan. Karena sel memori dikendalikan oleh 3 gerbang - gerbang input, lupa dan keluaran. Sel memutuskan informasi mana yang disimpan dan kapan unit mengakses informasi tersebut berdasarkan apakah operasi di gerbang terbuka atau tidak. LSTM memecahkan masalah ketergantungan jangka panjang yaitu LSTM berkinerja baik dalam kasus di mana informasi jalan kembali di masa lalu diperlukan untuk membuat prediksi masa depan. Jaringan saraf berulang dua arah atau *Bidirectional Recurrent Neural Network* (BRNN) melakukan pelatihan secara maju dan mundur ke dua jaringan syaraf tiruan (JST) berulang yang terpisah. Kedua JST tersebut kemudian dihubungkan ke lapisan luaran yang sama [16].

LSTM memiliki input x(t) yang dapat berupa output dari CNN atau urutan input secara langsung. h(t-1) dan c(t-1) adalah input dari LSTM pada waktu sebelumnya. Variabel o(t) adalah output dari LSTM untuk waktu saat ini. LSTM juga membangkitkan c(t) dan h(t) untuk dimasukkan ke LSTM pada langkah waktu berikutnya.



Gambar 1: Sel LSTM

Perhitungan untuk masukan dan luaran dalam satu langkah waktu tunggal adalah sebagai berikut [16]:

$$f_t = \sigma_t\left(W_t \times X_t + U_f \times h_{t-1} + b_f\right) \quad (6)$$

$$\boldsymbol{i_t = \sigma_t(W_i \times X_t + U_i \times h_{t-1} + b_i)} \quad (7)$$

$$o_t = \sigma_t(W_o \times X_t + U_o \times h_{t-1} + b_o) \quad (8)$$

$$c'_t = \sigma_c(W_c \times X_t + U_c \times h_{t-1} + b_c) \quad (9)$$

$$c_t = f_t . c_{t-1} + i_t . c'_t \quad (10)$$

$$h_t = o_t . \sigma_t(c_t) \quad (11)$$

2.5. Data Set

Proses pengumpulan data dilakukan dengan observasi dokumen dan wawancara dengan staf Prodi di Institut Bisnis dan Informatika Kosgoro 1957. Yang dicari adalah informasi mengenai hal-hal yang berkaitan dengan aktivitas akademik secara umum. Setelah dikumpulkan, kemudian data-data ini disimpan dengan format $\{s_i , c_i\}^N_{i=1}$ dimana S adalah kalimat dan C adalah kelas maksud. Data-data ini disimpan dalam basis data MySQL terpisah dalam dua tabel yaitu tabel intent_class dan tabel intent_test. Kedua tabel ini mempunyai format sama dan terdiri dari tiga kolom seperti di Tabel *1*.

Tabel 1. Format dataset dalam basis data

Nama	Tipe Data	Deskripsi
id	int	Kunci utama yang bernilai berurutan mulai dari satu
kalimat	text	Kalimat yang akan diberikan label maksud
maksud	char(30)	Label maksud untuk kalimat

2.6. Tahapan Penelitian

Tahapan proses yang dilakukan dalam penelitian ini adalah sebagai berikut:

1. Pengumpulan data dalam bentuk teks.
2. Pra-pemrosesan teks.
3. Pembangunan model jaringan syaraf tiruan (JST).
4. Proses pelatihan JST yang akan menghasilkan model JST dengan bobot lengkap.
5. Pembangunan aplikasi *Android* dengan bahasa pemrograman python.
6. Pembangunan aplikasi *back-end* dengan python dan diimplementasikan di komputer server.

3. HASIL DAN PEMBAHASAN

3.1. Pra-pemrosesan teks

Data diambil dari basis data kemudian dilakukan beberapa pra-proses sebelum dimasukkan ke JST. Data latih terdiri dari 464 baris dan data uji coba terdiri atas 255 baris. Jumlah kelas maksud yang akan diidentifikasi secara unik berjumlah 25 kelas maksud. Kelas-kelas ini dapat dilihat pada Tabel 2

Tabel 2: Rincian kelas maksud untuk kalsifikasi maksud

no	topik	kelas
1	Kata menyapa	greeting
2	Kuliah	cara_kuliah
3	Berhenti Kuliah	cara_berhenti_kuliah
4	Semester	def_semester
5	IPK dan cara penilaian	def_ipk
6	Nilai ujian dan total nilai jelek	def_nilai_jelek
7	Nilai ujian dan total nilai baik	def_nilai_bagus
8	Jadwal Kuliah	jadwal_kuliah
9	Ujian	cara_ujian
10	Ujian	def_ujian
11	Ujian Tengah Semester	cara_uts
12	Ujian Tengah Semester	def_uts
13	Ujian Akhir Semester	cara_uas
14	Ujian Akhir Semester	def_uas
15	Pengertian SIAKAD	def_siakad
16	Pengertian kuliah online dan caranya	kuliah_online
17	Bagaimana cara masuk SIAKAD	login_siakad
18	KKP	apa_kkp
19	KKP	cara_kkp
20	Skripsi	apa_skripsi
21	Skripsi	cara_skripsi
22	Remedial/ mengulang	apa_remedial
23	Remedial/ mengulang	cara_remedial
24	Berapa lama dibutuhkan kuliah sampai lulus	lama_kuliah
25	Apa saja yang dibutuhkan mahasiswa agar bisa lulus	syarat_lulus

Berikut ini pra-proses yang dilakukan:

1. Transformasi kata dibuat dalam huruf kecil semua.
2. Melakukan proses *steeming*, proses transformasi kata menjadi kata dasarnya (*root*). Proses ini menghilangkan semua imbuhan kata (*affixes*) seperti awalan kata (*prefixes*), sisipin kata (*infixes*), akhiran kata (*suffixes*) dan

menghilangkan awalan dan akhiran kata (*confixes*).

3. Data diubah menjadi larik dan disimpan dalan pengubah. Data latih disimpan dalam pengubah*kalimat* dan *maksud*. Data uji cobadisimpan dalam pengubah *kalimat_test* dan *maksud_test*.
4. Proses tokenisasi, yaitu proses pemisahan teks tertentu menjadi potongan-potongan lebih kecil yang disebut token.
5. Pembuatan indeks kosakata berdasarkan frekuensi kata tersebut dimulai dari kata yang paling sering muncul. Jadi jika nilai indeks lebih rendah berarti kata tersebut sering muncul.
6. Penambahan nilai 0 di awal urutan kata agar memastikan bahwa semua urutan dalam daftar memiliki panjang yang sama, proses ini dikenal dengan *padding sequences*.
7. Tansformasi kata-kata ke dalam nilai numerik kategorikal.
8. Kemudian dari numerik kategorikal, ditransformasi menjadi nilai biner 1 dan 0.
9. Pembacaan kata-kata yang akan disematkan dari file id.tsvlalu disimpan dalam pengubah *embedding_words* dan *embedding_index*.

3.2. Permodelan JST

Jaringan syaraf tiruan LSTM yang dibentuk terdiri dari beberapa lapisan. Berikut ini dijabarkan dari lapisan paling pertama sampai lapisan akhir.

1. Lapisan Pertama adalah vektor kata untuk penyematan.
   model.add(Embedding(num_words, 300, trainable=False,input_length=train_sequences .shape[1], weights=[embedding_matrix])).
   num_words : jumlah kata unik dari data pelatihan. Argumen kedua menunjukkan ukuran vektor penyematan yaitu 300. Argumen *input_length*, menentukan ukuran setiap urutan input. Setelah JST dilatih, sehingga bisa mendapatkan bobot lapisan penyematan dan dapat dianggap sebagai tabel yang digunakan untuk memetakan bilangan bulat ke vektor penyematan.
2. Lapisan kedua adalah *Bi-Directional LTSM.*
   model.add(Bidirectional(LSTM(300, return_sequences=True, recurrent_dropout=0.1, dropout=0.1), 'concat')).
   Lapisan ini merupakan model JST LSTM dengan 300 input sesuai dengan panjang vektor kata. *Dropout* adalah teknik pemilihan neuron secara acak untuk diabaikan selama pelatihan. Tujuannya adalah untuk menghindari *over-fitting*. Disini akan diabaikan 10% neuron selama pelatihan.
3. Lapisan ketiga *Drop Out.*
   model.add(Dropout(0.3))
   Lapisan ini mengabaikan 30% neuron secara acak di lapisan ini.
4. Lapisan keempat LSTM.
   model.add(LSTM(300,return_sequences=False , recurrent_dropout=0.1, dropout=0.1))
   Lapisan ini berupa mode JST LTSM kedua.
5. Lapisan kelima *Drop Out.*
   model.add(Dropout(0.3))
   Lapisan ini mengabaikan 30% neuron secara acak di lapisan ini.
6. Lapisan keenam adalah fungsi aktivasi.
   model.add(Dense(150, activation='relu'))
   Lapisan ini menunjukkan 150 unit dalam lapisan tersembunyi ini. Fungsi aktivasi yang digunakan adalah aktivasi ReLU dapat dinyatakan dengan f(x)=max(0,x) artinya nilai kurang dari 0 akan memotong sinyal input yang memiliki [17].
7. Lapisan ketujuh luaran dengan fungsi aktivasi.
   model.add(Dense(classes.shape[0], activation='softmax')).

3.3. Pelatihan JST

Pada penelitian ini pelatihan data diulang sebanyak 10, 20, 30, 40, 50, 60, 70, 80, 90, 100 epochs. Jumlah epoch adalah hyperparameter yang menentukan berapa kali algoritma pembelajaran akan bekerja melalui seluruh dataset pelatihan. Dalam satu epoch setiap sampel dataset pelatihan memiliki kesempatan untuk memperbarui parameter model internal. *Epoch* terdiri dari satu atau lebih batch, dalam pelatihan ini ditentukan 64 batch untuk setiap *epoch*.

Model dikompilasi dengan tiga jenis pengoptimal *Adaptive Moment (adam), Root Mean Square Propagation* (RMSprop) dan *Stochastic Gradient Descent* (SGD). Kompilasi fungsi kerugian menggunakan *categorical crossentropy* dan metrik validasi accuracy. *Accuracy* adalah selisih dari nilai yang dibaca benar oleh sistem dengan nilai yang memang benar sesungguhnya [6]. Fungsi categorical *cross entropy loss* adalah fungsi optimasi yang digunakan dalam kasus pelatihan model klasifikasi yang mengklasifikasikan data dengan memprediksi probabilitas apakah data milik satu kelas atau kelas lain.

Tabel 3. Metrik Accuracy dengan optimizer Adam dan RMSPO

Epochs	Adam		RMSprop	
	Accuracy(%)	Loss(%)	Accuracy(%)	Loss(%)
10	70.04	77.04	76.74	59.30
20	94.83	15.78	89.46	29.11

30	96.77	6.68	98.21	4.79
40	95.47	9.72	98.81	2.18
50	98.49	4.00	98.81	3.58
60	97.41	7.18	98.81	1.58
70	98.01	5.43	99.01	1.59
80	98.61	2.72	98.01	4.77
90	99.20	1.34	99.20	1.47
100	98.92	1.82	99.01	1.40

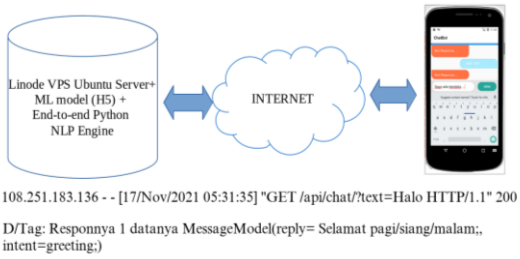
Tabel 4: Metrik Accuracy dengan optimizer SGD

Epochs	SGD	
	Accuracy(%)	Loss(%)
10	6.36	3.2436x102
20	5.17	3.2376x102
30	5.77	3.2291x102
40	7.55	3.2264x102
50	5.96	3.2168x102
60	6.56	3.2112x102
70	7.36	3.2062x102
80	7.55	3.1968x102
90	8.75	3.1857x102
100	12.92	3.1044x102

Dapat dilihat nilai akurasi berangsur naik ketika nilai epochs semakin besar pada optimizer adam dan RMSprop. Dengan optimizer adam, nilai akurasi dengan jumlah epochs 10 bernilai 70.04% dan kemudian naik drastis sebesar 94.83% pada epochs 20. Pada epochs 30-40 nilai akurasi menaik dan sampai 98.49%. Setelah itu, pada epochs 50-100 nilai akurasi terlihat tidak ada perubahan drastis dengan nilai tertinggi 99.20%. Dengan optimizer RMSprop, nilai akurasi pada awal epochs 10 bernilai 76.74% dan kemudian naik drastis sebesar 89.46% pada epochs 20. Pada epochs 30-100 nilai akurasi terlihat tidak ada perubahan drastis dengan nilai tertinggi 99.20%. Berbeda dengan adam dan RMSprop, optimizer SGD memperlihatkan nilai akurasi yang sangat kecil dibandingkan dengan optimizer sebelumnya. Nilai akurasi tertinggi dengan *optimizer* ini 12.92%.

Luaran dari pelatihan JST ini akan menjadi masukan untuk aplikasi *FAQ chatbot* berbasis *android*. Luaran ini terdiri dari empat file yaitu:

1. Model pembelajaran mesin yang disimpan dengan nama file *model_ML.h5.*
2. Semua kelas maksud yang unik disimpan dengan nama file *kelasMaksud.pkl*.
3. Hasil tokenisasi dengan nama file *tokenisasi.pkl*.
4. Hasil pemetaan urutan sumber dengan panjang variabel ke vektor dengan nama file *label_untuk_encoder.pkl*

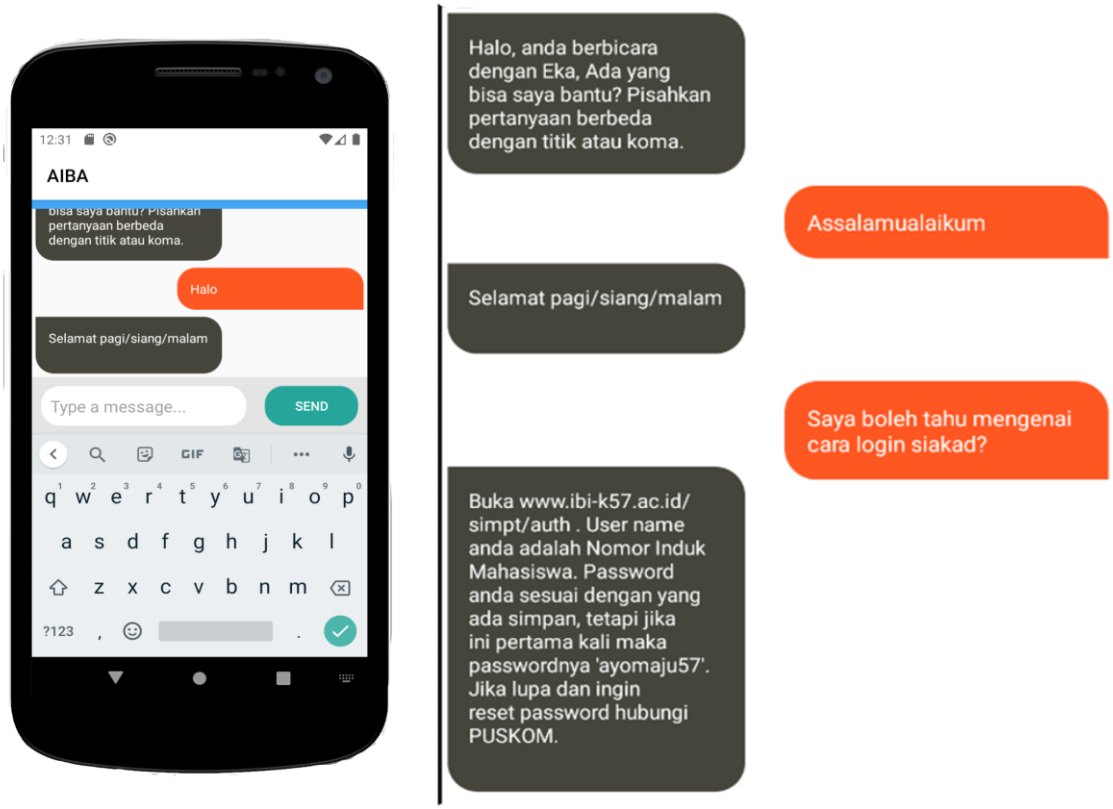


Gambar 2: Arsitektur *Client Server* Aplikasi *ChatBot* dengan *Cloud Server Linode*

3.4. Perancangan Aplikasi Berbasis *Android* dan Aplikasi *Back End* di server.

Dibutuhkan dua aplikasi, yaitu aplikasi untuk pengguna akhir dan aplikasi *back end* untuk menjalankan proses klasifikasi dengan model JST. Aplikasi pengguna akhir dibangun khusus hanya untuk telepon genggam dengan sistem operasi *Android*. Bahasa pemrograman yang digunakan untuk aplikasi ini adalah kotlin. Kebutuhan fungsional yang harus dipenuhi oleh aplikasi *FAQ chatter bot* ini adalah:

1. Login aplikasi dengan masukan teks nama dan jurusan.
2. Aplikasi dapat menerima pertanyaan dari pengguna ke *ChatBot* melalui teks pesan.
3. *Chatbot* menerima pertanyaan tersebut kemudian melakukan koneksi ke *Cloud Server* dan mengirimkann teks pertanyaan melalui melalui API (*Application Programming Interface*).
4. *Chatbot* menerima jawaban dari server dan menampilkannya di kotak dialog.



Gambar 3: Aplikasi *ChatBot FAQ* untuk IBI-K57

Aplikasi *back end* di server adalah aplikasi dengan kode python yang akan memasukkan input ke model JST *model_ML.h5* dan mengembalikan luaran yaitu kelas maksud yang terindetifikasi, probabilitas kelas tersebut dan respon yang sesuai dengan kelas maksudnya. Kebutuhan fungsional yang harus dipenuhi oleh aplikasi di sisi server adalah:

1. *Cloud Server* menerima pertanyaan dari aplikasi pengguna dan kemudian menjalankan kode python untuk mencoba mencari apa maksud dari pertanyaan tersebut. Kalimat pertanyaan menjadi masukan ke model JST LSTM agar bisa dicari apa maksud pertanyaan tersebut. File yang digunakan di proses adalah *model_ML.h5, kelasMaksud.pkl, tokenisasi.pkl* dan *label_untuk_encoder.pkl.*
2. Server dapat menemukan kelas maksud untuk pertanyaan tersebut atau pun tidak menemukannya. Nilai ini ditentukan dari seberapa besar probabilitas kelas maksud yang teridentifikasi. Disini akan ditetapkan nilai batasan 80% untuk nilai probabilitas.

a. Jika ditemukan berarti nilai probabilitas >= 80%, maka server akan mencari pasangan jawaban untuk kelas tersebut sebagai respon ke aplikasi pengguna akhir.

b. Jika tidak ditemukan berarti nilai probabilitas < 80%, maka server akan mengembalikan pesan khusus. Pesan khusus ini sejenis dengan peryataan berikut *"Maaf, saya tidak memahami pertanyaan anda, bisa ulangi lagi?", "Saya tidak menemukan jawaban untuk pertanyaan ini. Pertanyaan ini akan saya simpan"*. Kasus-kasus seperti ini akan menjadiumpan balik. Pertanyaan-pertanyaan yang tidak dimengerti tersebut dimasukan ke dalam tabel *intent_baru* di MySQL. Data-data pada tabel *intent_baru* ini akan diberikan label oleh teknisi data dan dimasukkan lagi ke proses pelatihan selanjutnya. Sehingga jika semakin banyak pertanyaan maka akan memperbanyak kosa kata baru untuk mesin. Banyaknya kosa kata akan membuat mesin semakin pintar. Ini merupakan ciri dari proses *supervised learning*.

3. *Server* mengirimkan hasil respon terdiri dari kelas maksud yang terindentifikasi beserta pasangan jawabannya ke aplikasi pengguna akhir melalui API. Hasil respon ini disimpan dalam format *JSON*.

Di sisi server dibutuhkan aplikasi yang dapat menjalankan model JST kami. Karena model JST kami dibangun dengan bahasa pemrograman python maka kami membutuhkan kerangka aplikasi yang dapat menjalankan python. *Flask framework* adalah salah satu *web framework* yang dapat kami jadikan sebagai lapisan aplikasi di sisi server seperti yang dilakukan dalam penelitian [18][19].

Tabel 5. Contoh pertanyaan dan klasifikasi maksud yang dilakukan JST

Question &Kelas Maksud	Probabilitas
Question: "Assalamualakum, saya mau tanya mengenai pelaksanaan UTS" **Kelas Maksud**: 'cara_uts'	89.74%
Question: "Saya pengen tahu masalah cara pelaksanaan ujian bu" **Kelas Maksud**: 'cara_ujian'	99.99%
Question: "saya mau mengisi pulsa dong" **Kelas Maksud**: 'apa_remedial' Probabilitas dibawah 80% maka akan dikembalikan pesan khusus	39.76%

Berikut ini adalah logika untuk program aplikasi *back end* di server:

1. Load file *model_ML.h5, kelasMaksud.pkl*, *tokenisasi.pkl* dan *label_untuk_encoder.pkl.*
2. Terima teks masuk dari aplikasi.
3. Lakukan proses *stemming*.
4. Lakukan proses tokenisasi.
5. Lakukan proses *pad-sequences*.
6. Kembalikan kelas maksud dengan nilai probabilitasnya.
7. Jika probabilitas >= 80% maka kembalikan pasangan jawaban untuk pertanyaan tersebut.
8. Jika probabilitas < 80% maka kembalikan pesan khusus dan simpan pertanyaan tersebut dalam tabel *intent_test*.

3.5. Uji Coba aplikasi

Pengujian aplikasi dilakukan dengan teknik uji coba *black-box*. Pengujian kotak hitam adalah pengujian berdasarkan spesifikasi persyaratan tanpa perlu memeriksa kode program [20]. Pengujian ini dilakukan berdasarkan sudut pandang pengguna akhir dengan kumpulan kasus uji coba yang dapat diprediksi. Tiga jenis kasus yang dicoba adalah (1) koneksi terputus atau ada kendala teknis, (2) Pertanyaan yang relevan dengan kelas maksud, (3) Pertanyaan yang tidak relevan dengan kelas maksud atau *Out Of Topic*. Setiap kasus diujikan dengan minimal 10 kali untuk kasus 1 dan 20 kali untuk kasus 2 dan 3.

Tabel 6. Kasus-kasus uji coba yang digunakan

Kasus Uji Coba	Respon
Jika koneksi terputus atau ada kendala teknis	Pesan pemberitahuan "sepertinya koneksi sedang ada kendala" di aplikasi pengguna
Pertanyaan yang relevan dengan kelas maksud	Aplikasi mengembalikan kelas maksud yang benar dengah probabilitas >= 80% beserta jawaban yang sesuai
Pertanyaan yang tidak relevan dengan kelas maksud atau *Out Of Topic*	Aplikasi mengembalikan kelas maksud yang salah dengah probabilitas < 80% dan pesan khusus. Kemudian pertanyaan tersebut disimpan dalam basis data.

4. KESIMPULAN

Penelitian ini bertujuan membuat hal yaitu, (1) pembuatan *FAQ chatbot* untuk menjawab pertanyaan-pertanyaan akademik umum, (2) implementasi model ini ke dalam aplikasi yang bisa digunakan.

Pembuatan *chatbot* digunakan proses pembelajaran mesin dengan jaringan syaraf tiruan LSTM. Ditambahkan vektor kata sebagai penyematan kata dari korpus Wikipeda yang sudah dilatih dan dibagi dengan lisensi publik. Hasilnya adalah model *chatbot* yang dapat memahami maksud dari sebuah kalimat. Metrik kinerja pembelajaran mesin yang digunakan adalah pengoptimal *Adaptive Moment* (adam), fungsi kerugian *categorical crossentropy* dan *accuracy*. Dari pelatihan didapatkan nilai accuracy sebesar 98.92% dan nilai RMSE (*Root Mean Square Error*) sebesar 2.18%. Untuk tujuan kedua yaitu implementasi, dibangun dalam arsitektur *client-server*. Di sisi *client* dibangun aplikasi telepon gengam berbasis *Android* dengan bahasa pemrograman kotlin. Di sisi *server* dibangun aplikasi dengan kerangka web *Flask*. Ada beberapa alasan kenapa *Flask* bisa digunakan untuk aplikasi *back-end*. Yang pertama dan yang paling penting, dia bisa menjalankan kode python. Kedua dia bisa berjalan sebagai aplikasi website yang bisa diakses di jaringan komputer publik.Pengujian aplikasi menggunakan uji coba *black box* dilakukan oleh pengembang dan mahasiswa sebagai tester.

Penelitian selanjutnya akan mengembangkan *chatbot* agar dapat memperbaharui mode JST tanpa harus melakukan pelatihan dari awal agar lebih adaptif terhadap kosa kata baru.

5. DAFTAR PUSTAKA

[1]. Y. Sumikawa, M. Fujiyoshi, H. Hatakeyama, and M. Nagai. 2019. "Supporting Creation of FAQ Dataset for E-Learning Chatbot"., *in*

Intelligent Decision Technologies 2019, pp. 3–13.

[2]. S. Ayanouz, B. A. Abdelhakim, and M. Benhmed. 2020. "A Smart Chatbot Architecture based NLP and Machine Learning for Health Care Assistance"., NISS2020: *The 3rd International Conference on Networking, Information Systems & Security*, pp 1-7. Doi: 10.1145/3386723.3387897.

[3]. F. Sethi. 2020. "FAQ (Frequently Asked Questions) ChatBot for Conversation"., *International Journal of Computer Sciences and Engineering*, vol. 8, no. 10, 2020. Doi: 10.26438/ijcse/v8i10.710.

[4]. T. Lalwani, S. Bhalotia, A. Pal, S. Bisen, and V. Rathod. 2018. "Implementation of a Chat Bot System using AI and NLP"., *International Journal of Innovative Research in Computer Science & Technology*, vol. 6, no. 3, pp. 26–30, May 2018, doi: 10.21276/ijircst.2018.6.3.2

[5]. A. Onan and M. Tocoglu. 2021. "A Term Weighted Neural Language Model and Stacked Bidirectional LSTM Based Framework for Sarcasm Identification"., *IEEE Access*, vol. PP, p. 1, Jan. 2021, Doi: 10.1109/ACCESS.2021.3049734.

[6]. D. Tuhenay and E. Mailoa. 2021. "Perbandingan Klasifikasi Bahasa Menggunakan Metode Naïve Bayes Classifier (Nbc) Dan Support Vector Machine (Svm)"., *Jurnal Informatika dan Komputer*, vol. 4, no. 2, 2021. Doi: 10.33387/jiko.

[7]. A. Huddar, C. Bysani, C. Suchak, U. D. Kolekar and K. Upadhyaya. 2020. "Dexter the College FAQ Chatbot"., *2020 International Conference on Convergence to Digital World - Quo Vadis (ICCDW)*, pp. 1-5. Doi: 10.1109/ICCDW45521.2020.9318648.

[8]. K. A. Nugraha, and D, Sebastian. 2021. "Chatbot Layanan Akademik Menggunakan K-Nearest Neighbor"., Jurnal Sains dan Informatika, 7(1), 11-19.

[9]. D. Gunawan, F. P. Putri, and Meidia, H. 2020. "Bershca: bringing chatbot into hotel industry in Indonesia"., Telkomnika, 18(2), 839-845.

[10]. Y. D. Prabowo, H. L. Warnars, W. Budiharto, A. I. Kistijantoro, and Y. Heryadi. 2018. "Lstm and simple rnn comparison in the problem of sequence to sequence on conversation data using bahasa indonesia". *In 2018 Indonesian Association for Pattern Recognition International Conference (INAPR)* (pp. 51-56). IEEE.

[11]. S. Luzi and B. Robert. 2018. " Evaluating the Ability of LSTMs to Learn Context-Free Grammars". 115-124. 10.18653/v1/W18-5414.

[12]. B. Wang, A. Wang, F. Chen, Y. Wang, and C. C. J. Kuo. 2019. "Evaluating word embedding models: Methods and experimental results"., *APSIPA Transactions on Signal and Information Processing*, vol. 8. Cambridge University Press, 2019. Doi: 10.1017/ATSIP.2019.12.

[13]. J.-H. Wang, T.-W. Liu, X. Luo, and L. Wang. 2018. "An LSTM Approach to Short Text Sentiment Classification with Word Embeddings*"., Proceedings of 30th Conference of Computational Linguistics and Speech Processing (ROCLING)*, pp 214-233.

[14]. C. Xu, L. Xie, and X. Xiao. 2018. "A Bidirectional LSTM Approach with Word Embeddings for Sentence Boundary Detection"., *Journal of Signal Processing Systems*, vol. 90, no. 7, pp. 1063–1075, Jul. 2018. Doi: 10.1007/s11265-017-1289-8.

[15]. M. Aleedy, H. Shaiba, and M. Bezbradica. 2019. "Generating and Analyzing Chatbot Responses using Natural Language Processing"., *International Journal of Advanced Computer Science and Application (IJACSA)*, volume 10, issue 9, 2019. Doi: 10.14569/IJACSA.2019.0100910.

[16]. M. Schuster and K. K. Paliwal. 1997. "Bidirectional recurrent neural networks"., *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997. Doi: 10.1109/78.650093.

[17]. H. Akbar. 2021. "Klasifikasi Kanker Serviks Menggunakan Model Convolutional Neural Network (Alexnet)"., *JIKO* (*Jurnal Informatika dan Komputer)*, vol. 4, no. 1, 2021. Doi: 10.33387/jiko.

[18]. A. A. Ahmed and G. Agunsoye. 2021. "A real-time network traffic classifier for online applications using machine learning"., Algorithms, vol. 14, no. 8, Aug. 2021. Doi: 10.3390/a14080250.

[19]. A. Najwa Arba'in, S. Letchumy, and M. Belaidan. 2020. "Fault Detection And Prediction In The Semiconductor Manufacturing Process," *International Journal of Management (IJM)*, vol. 11, no. 11, pp. 2023–2028, 2020. Doi: 10.34218/IJM.11.11.2020.192.

[20]. S. Nidhra. 2012. "Black Box and White Box Testing Techniques - A Literature Review," *International Journal of Embedded Systems and Applications*, vol. 2, no. 2, pp. 29–50, Jun. 2012. Doi: 10.5121/ijesa.2012.2204.