Rancang Bangun *Chatbot* Jual Beli Online Berbasis *Support Vector Machine* dan *Normalized Levenshtein Distance*

Ahmad Arif Samudro[a,*] , Muchammad Andhika Supriyanto[b] , Rizky Muhamad Rasyid[c]

[abc]*Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya*

Abstrak

Dewasa ini hampir dari kita semua pernah atau bahkan sering melakukan transaksi jual beli barang secara *online.* Beberapa orang bahkan menjadikan kegiatan jual beli *online* sebagai pekerjaan tetap mereka. Kegiatan jual beli secara *online* menawarkan kemudahan untuk kita agar tidak perlu bersusah payah untuk membeli barang yang kita inginkan. Kondisi ini juga menuntut akan adanya kemudahan dalam mendapatkan informasi terkait barang yang diperjual belikan kapanpun dan dimanapun. Namun tidak semua penjual menawarkan layanan untuk menjawab pertanyaaan dari pelanggannya selama 24 jam setiap harinya. Oleh karena itu, dalam paper ini kami membuat sebuah *chatbot* jual beli *handphone* berbasis *Support Vector Machine* dan *Normalized Levenshtein Distance* yang diharapkan nantinya mampu untuk dapat membantu penjual menjawab setiap pertanyaan pelanggannya selama 24 jam.

Kata kunci: chatbot, SVM, Levenshtein Distance, Jual Beli Online

1. Pendahuluan

Sebuah *chatterbot* atau *chatbot* merupakan sebuah program computer yang di desain untuk mensimulasikan sebuah percakapan secara cerdas dengan satu atau lebih pengguna manusia melalui metode pendengaran ataupun tekstual. *Chatbot* dapat diprogram untuk obrolan ringan, atau juga dapat berfungsi sebagai media interaksi dengan pengguna, memberikan mereka jawaban berdasarkan pertanyaan biasa. *Chatbot* dapat memahami konteks serta memberikan tanggapan sesuai dengan pesan atau pertanyaan yang diberikan kepadanya. *Chatbot* adalah salah satu dari banyak contoh *Artificial Intelligent* yang awalnya dirancang untuk sarana hiburan dan beberapa diantaranya dirancang untuk lulus Uji Turing. [1]

Tidak dapat kita pungkiri bahwa hampir dari kita semua pernah bahkan sering melakukan transaksi atau jual beli barang secara *online.* Jual beli secara online memberikan kemudahan untuk kita tanpa harus susah payah keluar rumah untuk membeli barang yang diinginkan. Sudah banyak organisasi yang menyediakan portal jual beli online seperti Tokopedia, Lazada, Shopee, dan lainnya. Pada zaman teknologi yang serba canggih seperti sekarang ini, banyak orang yang ingin mendapatkan informasi secara cepat dimanapun dan kapanpun termasuk dalam konteks jual beli online. Pembeli ingin dengan cepat mengetahui informasi barang yang ingin mereka beli namun tidak semua penjual menawarkan layanan untuk mejawab informasi 24 jam setiap harinya.

Oleh karena itu, *chatbot* yang dibuat diharapkan mampu untuk dapat memudahkan pengguna mendapatkan informasi terhadap barang yang dijual oleh sang pembeli. Pengguna memiliki pilihan untuk mengobrol dengan bot dan mengajukan pertanyaan normal untuk mendapatkan tanggapan. *Chatbot* memiliki respon yang diprogram sebelumnya, tetapi dapat bekerja dengan informasi dinamis dari pesan pengguna untuk membuat percakapan yang relevan dan menjawab dengan informasi yang relevan. Ini adalah alternatif yang menjanjikan dibandingkan harus menunggu jawaban dari penjual yang tidak tahu kapan akan membalas. Dalam konteks ini, *chatbot* digunakan untuk

*Corresponding Author

memvisualisasikan isi korpus (contoh teks pada dunia nyata) dan untuk memberikan jawaban atas domain tertentu yaitu jual beli *handphone*.

2. Tinjauan Pustaka

Bagian ini menjelaskan teori yang digunakan untuk mengerjakan penelitian.

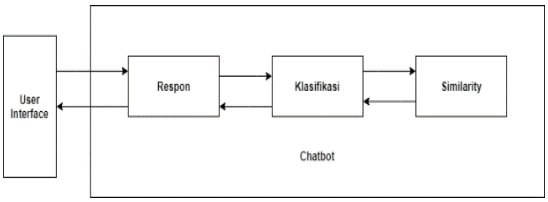
2.1 Chatbot

Secara harfiah chatbot terdiri dari dua kata yaitu chat dan bot. Chat sendiri dapat diartikan sebagai kegiatan komunikasi yang menggunakan sarana tulisan. Sedangkan bot merupakan program yang memiliki sejumlah data yang bila diberi input akan menghasilkan output sebagai jawaban. Chatbots dapat membantu dalam interaksi manusia dengan komputer dengan mengajukan pertanyaan dan menanggapi pertanyaan-pertanyaan pengguna [1]. Masukan ke program ini adalah teks bahasa alami atau Natural Language Processing (NLP), dan program harus memberikan jawaban yang merupakan jawaban terbaik untuk kalimat masukan. Proses ini diulang terus-menerus selama percakapan berlangsung dan responsnya berupa teks atau ucapan.

ELIZA yang diciptakan oleh Joseph Weizenbaum merupakan salah satu chatbot pertama yang diciptakan. Menciptakan Chatbot membutuhkan skill pemrograman yang sangat profesional dan pengembang berpengalaman untuk mendapatkan percakapan yang sangat realistis. Chatbot yang sempurna membutuhkan dataset yang sangat besar dan beragam agar dapat memberikan respone yang masuk akal untuk pertanyaan masukan pengguna. Dataset digunakan untuk menciptakan pengetahuan serta pembelajaran dari Chatbot itu sendiri. Belajar di sini berarti menyimpan kalimat baru dan kemudian menggunakannya nanti untuk memberikan jawaban yang tepat untuk kalimat serupa.

Chatbot dapat dibagi menjadi tiga bagian: Respons, klasifikasi dan similarity [2] (seperti yang ditunjukkan pada Gambar. 1), yang dijelaskan sebagai berikut:

- Respons: digunakan untuk penghubung antara masukan pengguna dengan klasifikasi serta mengatur input dan output
- Klasifikasi: penghubung antara respon dengan similarity. fungsi utamanya adalah melakukan preprocessing kalimat masukan, mengirim kalimat yang telah diproses ke similarity.
- Similarity: menyamakan masukan yang sudah dilakukan klasifikasi dengan dataset yang ada dan mengeluarkan output message dari masukan tersebut



Gambar 1. Komponen Chatbot

2.2 Support Vector Machine

Support Vector Machine (SVM) merupakan suatu teknik atau cara untuk melakukan prediksi baik dalam kasus regresi ataupun klasifikasi. Teknik SVM sendiri digunakan untuk mendapatkan fungsi pemisah *(hyperplane)* yang optimal untuk dapat memisahkan dua variabel yang berbeda dengan ukuran margin yang maksimal [3]. Prinsip dasar SVM adalah linear classifier, dan selanjutnya dikembangkan agar dapat bekerja pada problem non-linear. SVM bekerja dengan baik dengan data terstruktur dan semi-terstruktur seperti tulisan tangan (text), crawling data pada web pages, serta pendeteksian wajah. [4]

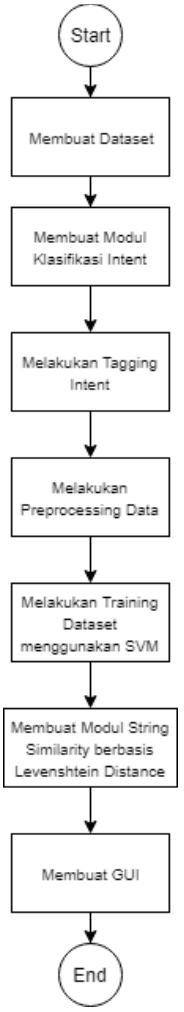
2.3 Normalized Levensthtein Distance

Levenshtein Distance adalah jumlah operasi minimum untuk mengubah satu *string* menjadi *string* yang lain [5]. Merupakan suatu satuan pengukuran perbedaan antara dua *string* yang diciptakan oleh Vladimir Levenshtein pada tahun 1965. Adalah jarak terkecil dari penyisipan, penghapusan, dan substitusi yang dibutuhkan untuk mengubah sebuah *string* menjadi *string* yang lain. Ini adalah sebuah *metric string distance* yang implementasinya menggunakan *dynamic programming* (Wagner–Fischer

algorithm), dengan hanya menggunakan 2 baris data. Sebuah kalimat dituliskan sebagai O(m) dan O(n), sehingga penulisan algoritmanya adalah O(m.n) [5]. Nilai dari *levensthein distance* dibagi dengan panjang string terpanjang disebut dengan *Normalized Levenshtein Distance*.

3. Metodologi

Berikut ini adalah metodologi yang digunakan sebagai acuan penulis untuk melakukan pembuatan *chatbot*. Gambar 2 merupakan alur metodologi pengerjaan penelitian penulis.



Gambar 2. Alur Metodologi Penelitian

3.1 Membuat Dataset

Kami melakukan dummy dataset dengan topik jual beli online handphone sebanyak 120 percakapan antara penjual dan pembeli. Dataset berisikan dua kolom, yaitu kolom marker dan kolom *message*. Dimana pada kolom marker terdiri dari pemilik *message* (penjual atau pembeli). Sedangkan pada kolom message terdiri dari *start* untuk awalan dari percakapan, isi percakapan (kumpulan message), dan *end* untuk akhiran dari satu percakapan. Dataset kami terdiri dari 600 turn dan 1200 row utterance pembeli dan penjual yang diisimpan dengan format *comma separated values (CSV)*.

3.2 Membuat Modul Klasifikasi Intent

Dataset yang telah dibuat sebelumnya dilakukan analisis untuk menentukan intent dari korpus dialog. Intent dikelompokkan dalam 9 kategori, dimana pada setiap intent memiliki skema intent yang jelas agar menjadi acuan melakukan data tagging pada tahapan selanjutnya. Sembilan buah klasifikasi skema tagging intent yaitu Stok Barang, Pengiriman, Harga, Konfirmasi, Garansi, Perbandingan HP, Spesifikasi, Kondisi Barang, dan Lainnya.

3.3 Melakukan Tagging Intent

Disini kami melakukan pemberian tag pada setiap *message* penjual yang ada pada dataset kami dan menentukan termasuk pada klasifikasi intent yang mana. Tagging dilakukan sebanyak tiga kali oleh orang yang berbeda, sehingga didapatkan kesimpulan dari tagging dengan cara mengambil modus tagging. Jika tidak ada modus pada utterance maka dilakukan diskusi untuk mengambil kesimpulan.

3.4 Melakukan Preprocessing Data

Agar data dapat digunakan oleh algoritma machine learning, kita harus memrosesnya terlebih dahulu dan mengonversinya menjadi format yang tepat. Sehingga, kami melakukan tahapan preprocessing data untuk membersihkan data seperti menghilangkan tanda baca, menghilangkan angka, *case fold*, *tokenizing*, dan *stemming*. Hasil dari preprocessing data yang

sudah dilakukan, kemudian disimpan menjadi sebuah dataset baru yang berisi *list of dictionary*. Dimana terdapat tiga key yaitu *message, response, voting*.

3.5 Melakukan Training Dataset menggunakan SVM

Setelah dilakukan preprocessing data, data diubah menjadi vector menggunakan vectorizer dari sklearn dan dilakukan pembobotan menggunakan TF-IDF. Kami menggunakan 500 pertanyaan terkait dengan jual beli handphone yang sudah kami *tagging* secara manual sebelumnya. Data dibagi menjadi 80% untuk data training dan 20% untuk data testing. Untuk mendapatkan nilai *accuracy, precision,* dan *recall* yang terbaik, kami melakukan lima kali percobaan training dengan mengubah nilai C dan g dalam SVM. Hasil dari training dataset disimpan menjadi sebuah model baru yang bernama model.fp1.

3.6 Membuat Modul String Similarity berbasis Normalized Levenshtein Distance

Sebelum melakukan string similarty dengan menggunakan normalized levenshtein distance, pertama memuat model dan dataset yang telah disimpan dari tahap sebelumnya, kemudian memunculkan entri masukan. input yang dimasukkan oleh pengguna akan dipreprocessing terlebih dahulu untuk menghilangkan tanda baca dan angka, serta dilakukan casefold. Kemudian input diprediksi kategorinya berdasarkan model, lalu dari dataset diambil message yang memiliki kategori yang sama dengan prediksi input. Dilakukan perhitungan normalized levensthein distance terhadap input dengan seluruh message dengan kategori yang sama. Diambil dua nilai yang paling mendekati satu dan dilakukan random untuk mengeluarkan response.

3.7 Membuat GUI

Kami membuat GUI sederhana menggunakan library tkinter pada python. GUI digunakan untuk memudahkan visualisasi dari *chatbot* yang kami buat.

4. Hasil dan Pembahasan

4.1 Membuat Modul Klasifikasi Intent

Setelah melakukan analisis dari korpus dialog untuk menentukan intent dari 9 kategori yang telah di tentukan. Berikut hasil dari skema tagging intent dari semua intent yang ada. Tabel 1 adalah skema tagging intent Stok Barang.

Tabel 1. Skema Tagging Intent Stok Barang

Ujaran yang termasuk	**Ujaran yang tidak termasuk**
Mas redmi 5a masih ada kah?	gan yg 2GB ada?
ready warna apa saja mas?	gan yang silent camera ready?
tipe kevlar sudah ready gan?	
Ready black matte gan?	
s9+nya ready gan?	

Tabel 2 adalah skema tagging intent Pengiriman

Tabel 2. Skema Tagging Intent Pengiriman

Ujaran yang termasuk	**Ujaran yang tidak termasuk**
Kirim hari ini bisa?	tokonya di mana?
bsa gojek sekarang?	bisa kirim WA?
masih bisa kirim grab waktu sekarang?	
Ok tlong di packing yg rapi ya,. Soalnya buat hadiah mksh	
Bogornya mana gan? Bisa COD?	

Tabel 3 adalah skema tagging intent Harga

Tabel 3. Skema Tagging Intent Harga

Ujaran yang termasuk	**Ujaran yang tidak termasuk**
harganya bisa turun lagi kah?	gabisa di edit aja harganya?
samsung S9 sudah turun gan?	
gan mau grosiran kena brp nih?	
Sama ongkir jadi berapa?	

Tabel 4 adalah skema tagging intent Konfirmasi

Tabel 4. Skema Tagging Intent Konfirmasi

Ujaran yang termasuk	**Ujaran yang tidak termasuk**
Sdh di trf mksh	Oke saya mau yang s9+
oke nanti saya hubungi lagi bro	yg oke buat pubgm high ada gak?
sudah ya boss, pake gojek Hari ini	

Tabel 5 adalah skema tagging intent Garansi

Tabel 5. Skema Tagging Intent Garansi

Ujaran yang termasuk	Ujaran yang tidak termasuk
garansi apa mas?	masih segel kan?
barang kalau sudah diterima bisa di refund gan?	
bisa tuker kalau dapat yang jelek?	
ga bisa klaim garansi di indo dong ya?	
wah nggak TAM ya	

Tabel 6 adalah skema tagging intent Perbandingan HP

Tabel 6. Skema Tagging Intent Perbandingan HP

Ujaran yang termasuk	Ujaran yang tidak termasuk
xiaomi sama samsung bagus mana?	ini buat game oke gak?
A5 sama A2 bagus mana gan?	
kalau dibandingkan dengan z5?	
kameranya bagusan mana sm yg note?	
bagusan mana ini sama 1s?	

Tabel 7 adalah skema tagging intent Spesifikasi

Tabel 7. Skema Tagging Intent Spesifikasi

Ujaran yang termasuk	Ujaran yang tidak termasuk
Kameranya emang 3?	Warna merah?
ini yg Global version gan?	Kalau warna ijo lumut?
ini udah unlock bootloader ya mas?	
support 4g?	
ram brapa gan?	

Tabel 8 adalah skema tagging intent Kondisi Barang

Tabel 8. Skema Tagging Intent Kondisi Barang

Ujaran yang termasuk	Ujaran yang tidak termasuk
ini ori? Kok harganya segitu?	kok harganya murah banget ya?
ori gan?	bisa tuker kalau dapat yang jelek?
HP mulus?	
ori pabrik kw dong?	
hp lolos qc gan?	

Tabel 9 adalah skema tagging intent Lainnya

Tabel 9. Skema Tagging Intent Lainnya

Ujaran yang termasuk	Ujaran yang tidak termasuk
gambarnya ga ada yg lebih jelas?	Semua yang sudah termasuk di intent sebelumnya
syaratnya apa saja?	
bisa kirim WA ?	
oh oke deh, ordernya gimana?	
gabisa di edit aja harganya?	

4.3 Melakukan Tagging Intent

Setelah skema intent dibuat, pemberian *tag* dilakukan pada setiap *message* penjual yang ada pada dataset kami dan menentukan termasuk pada klasifikasi intent yang mana. Hasil dari *tagging* adalah 100 message termasuk kedalam intent Stok Barang, 64 message termasuk kedalam intent Pengiriman, 41 message untuk intent Harga, 123 message intent Konfirmasi, 33 message intent Garansi dan Spesifikasi, 14 message termasuk intent Perbandingan HP, 49 message termasuk intent Kondisi Barang dan 47 message tersisa masuk kedalam intent Lainnya. Tabel 10 menampilkan komposisi dari *tagging* setiap intent pada dataset kami.

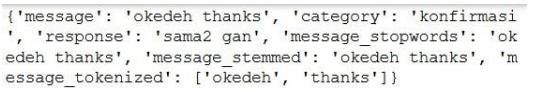
Tabel 10. Komposisi Tagging dari setiap Intent

Tag Intent	Jumlah
Stok Barang	100
Pengiriman	64
Harga	41
Konfirmasi	123
Garansi	33
Perbandingan HP	14
Spesifikasi	33
Kondisi Barang	49
Lainnya	47

4.4 Melakukan Preprocessing Data

Setelah melakukan *tagging*, agar data dapat digunakan oleh algoritma machine learning, kami melakukan tahapan preprocessing data untuk membersihkan data seperti menghilangkan tanda baca, menghilangkan angka, *case fold, tokenizing*, dan *stemming*. Gambar 3 merupakan

salah satu hasil dari preprocessing *message* yang sudah kami lakukan.



Gambar 3. Salah satu Hasil dari Data Preprocessing

Hasil preprocessing dari sebuah *message* 'okedeh thanks', dimasukan ke dalam *category* 'Konfirmasi' dengan *respon* 'sama2 gan'. Dan untuk *stopwords, tokenizing, dan stemming* sudah dilakukan secara otomatis menggunakan vectorizer pada library sklearn sebelum melakukan training.

4.5 Melakukan Training Dataset menggunakan SVM

Dari data yang sudah dilakukan preprocessing pada tahap sebelumnya. Kami membagi data kami menjadi 80% untuk data training dan 20% untuk data testing. Untuk mendapatkan nilai *accuracy, precision*, dan *recall* yang terbaik, kami melakukan lima kali percobaan training dengan mengubah nilai C dan g dalam SVM. Tabel 11 memperlihatkan hasil *accuracy, precision*, dan *recall* dari setiap percobaan kami.

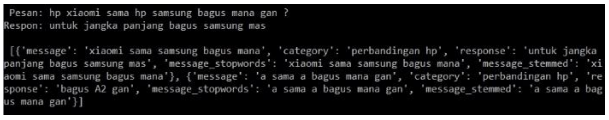
Tabel 11. Accuracy, Precision, Recall dari setiap percobaan

	Accuracy	Precision	Recall	C	g
Percobaan 1	0,267	0,029	0,111	1	Auto
Percobaan 2	0,594	0,543	0,385	0,5	1
Percobaan 3	0,703	0,778	0,581	1	1
Percobaan 4	0,693	0,809	0,577	1	1,2
Percobaan 5	0,693	0,774	0,577	1,1	1,2

Dari kelima percobaan yang telah kami lakukan, percobaan ketiga membuahkan hasil yang paling baik dengan *accuracy* sebesar 70,3%, kemudian *precision* sebesar 77,8%, dan *recall* sebesar 57% dengan parameter C=1 dan g=1.

4.6 Membuat Modul String Similarity berbasis Levenshtein Distance

Setelah melakukan training data, kami membuat modul string similaritynya. Gambar 4 dan 5 merupakan hasil dari percobaan modul similarity yang sudah kami buat. Pada gambar 4 dapat dilihat ketika ada input 'hp xiaomi sama hp samsung bagus mana gan?', model akan mencarikan dua buah pertanyaan yang paling mirip pada korpus kemudian memberikan respon secara random berdasarkan dua pertanyaan yang dianggap mirip tersebut. Dalam hal ini, model akan memberikan respon yaitu 'untuk jangka panjang bagus samsung mas' atau 'bagus A2 gan'.



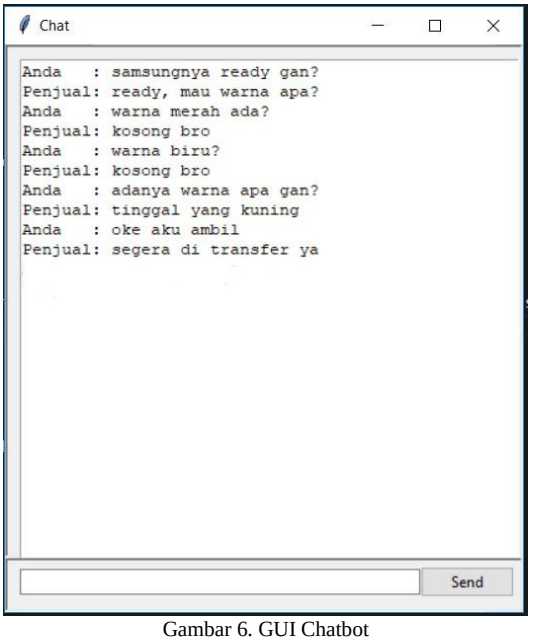
Gambar 4. Hasil String Similarity 1

Pada gambar 5 dapat dilihat ketika ada input 'warna merah ready gan?', model akan memberikan respon yaitu 'kosong bro' atau 'warna purple ready kak'.

Gambar 5. Hasil String Similarity 2

4.7 Membuat GUI

GUI yang kami buat memiliki sebuah *frame* untuk melihat percakapan antara penjual dan pembeli, lalu sebuah *textbox* untuk melakukan *input* pertanyaan, serta *button send* untuk menampilkan input dari *textbox* ke *frame*. Gambar 6 merupakan GUI chatbot yang sudah kami buat.



Gambar 6. GUI Chatbot

5. Kesimpulan

Chatbot yang kami buat dapat menjadi solusi untuk setiap penjual barang apapun khususnya handphone secara online untuk menyelesaikan masalah terkait ketersediaan pelayanan pemberian informasi yang dapat diberikan. *Chatbot* berbasis *Support Vector Machine* dan *Normalized Levenshtein* yang dibuat dapat memberikan response dari masukan dari pengguna. Seperti halnya saat pengguna menanyakan ketersediaan barang, akan dijawab dengan response yang ada pada tagging stok barang.

Namun, *Chatbot* yang dibuat pada paper ini masih memliki kekurangan, yang pertama adalah hasil *accuracy* sebesar 70,3% dirasa masih sangat kurang untuk membuat *chatbot* dapat menjawab pertanyaan secara benar dan konsisten. Kurangnya jumlah data dan kategori *tagging* yang kurang beragam masih menjadi halangan untuk dapat menaikkan *accuracy*. Hal tersebut berdampak pada saat mengeluarkan response yang kami random dari similarity antara masukan pengguna dengan message yang paling mendekati. Dikarenakan similarity antar message yang masih jauh, membuat response random yang kami terapkan membuat response yang keluar terlihat tidak seperti yang diharapkan (dapat dilihat pada gambar. 4 dan 5).

Selain itu terdapat saran yang bisa disarankan untuk pembuatan chatbot berikutnya terkait dengan apa yang telah dilakukan adalah (1) membuat dataset yang lebih banyak; (2) tagging yang lebih beragam; (3) memiliki data yang tidak terlalu beragam atau lebih memfokuskan data sesuai dengan taging intent yang diinginkan, agar mendapatkan response yang lebih baik

6. Daftar Rujukan

[1] S. Gupta, D. Borkar, C. D. Mello and S. Patil, "An E-Commerce Website based Chatbot," *International Journal of Computer Science and Information Technologies*, vol. 6, no. 2, pp. 1483-1485, 2015.

[2] S. A. A. Kader and J. Woods, "Survey on Chatbot Design Techniques in Speech Conversation Systems," *International Journal of Advanced Computer Science and Applications*, vol. 6, no. 7, pp. 72-80, 2015.

[3] Santosa, Data Mining, Teori dan Aplikasi, Jakarta: Graha Ilmu, 2007.

[4] X. Mu, X. Shen and J. Kirby, "Support Vector Machine Classifier Based on Approximate Entropy Metric for Chatbot Text-based Communication," *International Journal of Artificial Intelligence*, vol. 15, no. 2, pp. 1-16, 2017.

[5] [Online]. Available: https://xlinux.nist.gov/dads/HTML/Levenshtein.html. [Accessed 14 Desember 2018].

[6] [Online]. Available: https://github.com/luozhouyang/python-string-similarity#levenshtein. [Accessed 14 Desember 2018].