JISTech (Journal of Islamic Science and Technology)

JISTech, 6(2), 10-23, Juli-Desember 2021

ISSN: 2528-5718

http://jurnal.uinsu.ac.id/index.php/jistech

PENCARIAN ARTI AYAT AL-QUR'AN DENGAN SPEECH RECOGNITION MENGGUNAKAN ALGORITMA BERRY RAVINDRAN BERBASIS ANDROID

Hafizah Safwani[1], Suendri[2], Triase[3]

[1,2,3]Universitas Islam Negeri Sumatera Utara, Medan, Indonesia
Email : hafizah.safwani@uinsu.ac.id

ABSTRACT

Students in grades 4-5 Elementary School Al-Ikhlas need an Android-based application to findthe meaning of the Qur'anic verses to help and make it easier for students to find the meaning of the verses of the Qur'an and do the tasks given by Islamic Religion Teachers. Making the application's search for meaning verses of Qur'an using methods matching string (string matching) that is implemented in the search process said. String matching will perform the search process for a string or several strings found in a text or string. The algorithm used in string matching is berry ravindran, this algorithm has 2 phases, namely the preprocessing phase and the search phase. The phase preprocessing serves to create a shift value that will beused in the search phase. The value of the search phase is obtained from the rules of the algorithm. The Algorithm is berry ravindran implemented to perform analysis string matching in the application of finding the meaning of the verses of the Qur'an. This application to search for the meaning of the verses of the Qur'an can search for the meaning of the verses of the Qur'an by using voice, it is hoped that this voice search feature can facilitate the search for themeaning of the verses of the Qur'an. Speech recognition is implemented using one of the Google Cloud Platforms, namely the Google Speech API to detect spoken words when searching, so that it can make it easier for users to search for the meaning of the verses of the Qur'an. It is hoped that this application can be used as a source of knowledge and useful for grade 4-5 students at Elementary School Al-Ikhlas so that they are more diligent in reading theQur'an and finding out the meaning of the verse, in order to guide them to the right path to thepath that is pleasing to Allah.

Keyword**:** *String matching, Berry Ravindran, Speech Recognition, Android*

PENDAHULUAN

String Matching merupakan bagian penting dari proses pencarian *string* dalam sebuah dokumen. Hasil dari pencarian *string* dalam dokumen tergatung dari teknik dan cara yang digunakan(Ernawati et al., 2019). Penggunakan algoritma *Berry Ravindran* dalam menganalisis *string matching,* dikarenakan algoritma *Berry Ravindran* menggunakan berturut-turut dua karakter dari posisi *pattern* untuk pencarian nilai *shift*, sehingga mengurangi jumlah perbandingan karakter saat proses pencarian, hal tersebut dapat mempersingkat waktu pemrosesan *matching* dan mengurangi pemakaian memori yang diperlukan untuk nilai shift. Nilai *shift* didapat dari perhitungan dalam fase *preprocessing* berdasarkan karakter buruk *Berry Ravindran* (Siburian, 2017).

Salah satu *mobile device* yang memberikan pengadaan informasi adalah Android, Android memberikan berbagai layanan, salah satunya layanan pengenalan suara manusia (*Speech Recognition*). *Speech Recognition* merupakan proses identifikasi suara berdasarkan kata yang diucapkan berupa sinyal akustik yang ditangkap oleh *audio device*, parameter yang dibandingkan adalah tingkat penekanan suara yang kemudian dicocokan dengan *template database* yang tersedia (Nada et al., 2019). Aplikasi pencarian arti ayat al-qur'an ini dapat mencari arti ayat al-qur'an dengan menggunakan suara (*voice*), diharapkan dengan adanya fitur pencarian menggunakan suara dapat mempermudah pencarian arti ayat al-qur'an.

TINJAUAN PUSTAKA

String Matching

String matching adalah Susunan angka dan huruf yang biasanya direpresentasikan sebagai struktur data array. Pencocokan *string* penting dalam proses pencarian *string* dalam sebuah dokumen. Hasil pencarian *string* bergantung pada teknik dan metode pengumpulan *string* yang digunakan. *String* dapat berupa kata, frasa, atau kalimat. Hasil pencarian *string* tergantung pada metode pencocokan dan metode yang digunakan(Mukaromah & Amelia, 2019).

Algoritma *Berry Ravindran*

Algoritma *Berry Ravindran* merupakan salah satu algoritma *string matching* yang ditemukan oleh T. Berry dan S. Ravindran pada tahun 1999. Algoritma *Berry Ravindran* merupakan perpaduan antara algoritma *Quick Search* dan algoritma *Zhu-Takaoka* yang cara kerjanya terdiri dari dua fase, yaitu fase *preprocessing* dan fase pencarian, pergeseran algoritma *Berry Ravindran* lebih cepat dengan melakukan perhitungan pergeseran dua buah karakter dari posisi *string matching*. Pada fase *preprocessing* algoritma *Berry Ravindran* ditingkatkan pada perhitungan pergeseran untuk pola bukan karakter teks. Sehingga hal tersebut akan mengurangiwaktu proses pencarian. Algoritma *Berry Ravindran* sering diimplementasikan dalam berbagaifungsi pencarian *string*, seperti *Find* and *Replace*, pencocokan DNA dan lainnya. Implementasi algoritma ini dapat dilakukan pada berbagai macam sistem operasi seperti *Android*(Fadillah, 2018).

Speech Recognition

Speech Recognition (Pengenalan Ucapan/Suara) adalah Proses mengidentifikasi suara berdasarkan bahasa lisan dengan mengubah sinyal akustik yang ditangkap oleh perangkat input suara. Kata-kata diubah menjadi sinyal digital dengan mengubah gelombang suara menjadi serangkaian angka, mencocokkannya dengan kode tertentu, dan mencocokkan pola yang tersimpan di perangkat. Hasil identifikasi bahasa lisan dapat ditampilkan secara tertulis atau dibaca oleh perangkat teknis. (Huda, 2019).

Google Speech API

API (Application Programming Interface) merupakan teknologi yang sedang dikembangkan oleh *Google*. *API* memfasilitasi pertukaran informasi/data antara dua atau lebihaplikasi *software*. *Google Speech API* adalah *framework* untuk pengenalan suara dan mengubahnya menjadi teks (*string*). Input suara di-*record* pada perangkat *smartphone* kemudian dikirim ke server *Google* yang memiliki tugas pengenalan dan mengubahnyamenjadi teks *Speech Recognition* atau *Google Speech* adalah

suatu API yang disediakan *Google* untuk mengenali suara dengan cara digitalisasi kata dan mencocokan sinyal digital tersebut dengan sebuah pola yang tersimpan di dalam *database Google* (Akbar et al., 2019)

Android adalah sistem operasi perangkat *mobile* berbasis linux yang mencakup *middleware* dan aplikasi kunci. Pembuatan aplikasi pada *platform* Android menggunakan bahasa pemograman Java.

METODE PENELITIAN

Metode penelitian menggunakan metode penelitian *Research* and *Development* (R&D).*Research* and *Development* merupakan tahap awal dan penelitian dengan melakukan R&D dan pengujian produk dan layanan untuk melihat seberapa sukses dan sesuai produk dan layanan tersebut. Metode R&D adalah metode penelitian yang digunakan untuk memproduksi produk dan menguji keberhasilan produk tersebut. Studi analisis digunakan untuk memproduksi produk ini. (Zakariah, dkk. 2020) Langkah-langkah yang dilakukan menurut metode penelitian *Research and Development* (R&D) adalah sebagai berikut: (1)Potensi dan Masalah (2)Pengumpulan Informasi (3)Desain Produk (4)Validasi Desain (5)Perbaikan Desain (6)Ujicoba Produk (7)Revisi Produk akhir (8) Uji Coba Pemakaian.

HASIL DAN PEMBAHASAN

Penerapan Algoritma *Berry Ravindran*

1. Fase *Preprocessing*

Pada fase *preprocessing* dilakukan perhitungan untuk nilai pergeseran saat fase pencarian. Pada fase ini, berlaku beberapa kondisi yang berfungsi untuk menetukan nilai pergeseran pada brBc (*Berry-Ravindran bad-character shift*). Nilai pergeseran ditentukan dari berikut.

$$\text{bRBc [a,b]} = \min \begin{Bmatrix} 1 & if\ x\ [m-1] = a \\ m-i & if x[i]x[i+1] = ab \\ m+1 & if x[0] = b \\ m+2 & otherwise \end{Bmatrix}$$

Keterangan: brBc = *Berry Ravindran Bad Character*

Shift

x = *Pattern*

m = Panjang *Pattern*

a, b = Karakter di dalam teks

Untuk menjelaskan proses inisialisasi dari algoritma *Berry Ravindran* dengan *pattern* M,A,N,I,S yang akan dicari pada string AMMA MANIS TAGHNAA.

1) Pada fase *preprocessing*, *pattern* diurutkan sesuai abjad untuk membuat tabel *berry ravindran bad-character shift*. *Pattern* memiliki karakter sebanyak 5 karakter, yaitu A,I,M,N,S dan ditambah simbol (*) yang merupakan karakter selain M,A,N,I,S. untukproses awal, nilai pergeseran bernilai m+2 untuk karakter [a,b].

Rumus: m+2	**Keterangan**: m adalah panjang
Penyelesaian:	
m+2 = 5+2 = 7	

Sehingga akan didapat nilai pergeseran seperti pada tabel berikut ini:

Tabel 1 Nilai Pergeseran Untuk Karakter [a,b] bernilai m+2

brBc	A	I	M	N	S	*
A	7	7	7	7	7	7
I	7	7	7	7	7	7
M	7	7	7	7	7	7
N	7	7	7	7	7	7
S	7	7	7	7	7	7
*	7	7	7	7	7	7

2) Pada tahap kedua nilai pada tabel diganti menjadi m+1 untuk setiap karakter b=x[0] artinya karakter b digantikan menjadi karakter awal pada *pattern* sehingga menjadi brBc [a][M] ketika a bernilai 0, maka brBc akan menjadi brBc [0][M].

Rumus: m+1	**Keterangan**: m adalah panjang
Penyelesaian:	
m+1 = 6	

Dengan demikian, nilai pergeserannya akan berubah seperti tabel

berikut:

Tabel 2 Nilai Pergeseran Untuk Karakter b=x[0] Bernilai m+1 (a)

brBc	A	I	M	N	S	*
A	7	7	6	7	7	7
I	7	7	7	7	7	7
M	7	7	7	7	7	7
N	7	7	7	7	7	7
S	7	7	7	7	7	7
*	7	7	7	7	7	7

Ketika nilai a=1 maka brBc akan berganti menjadi brBc [1][R] yang memiliki nilai pergeseran sama, yaitu m+1. Hal tersebut berlaku seterusnya sama nilai a=4, didapatnilai brBc [4][R] seperti pada tabel berikut:

Tabel 3 Nilai Pergeseran Untuk Karakter b=x[0] Bernilai m+1 (b)

brBc	A	I	M	N	S	*
A	7	7	6	7	7	7
I	7	7	6	7	7	7
M	7	7	6	7	7	7
N	7	7	6	7	7	7
S	7	7	6	7	7	7
*	7	7	6	7	7	7

3) Pada tahap 3, nilai pergeseran diganti menjadi m-i dimana i adalah perulangan, perulangan berlangsung ketika nilai i=0, maka karakter x[i] dan x[i+1]. Jika karakter x[i] adalah M dan karakter x[i+1] adalah A maka brBc menjadi brBc [M][A].

Rumus: m-i
Keterangan: m adalah panjang *pattern*, i adalah banyak perulangan
Penyelesaian: m-i = 5-0 = 5

Dengan ini nilai pergeseran pada tabel akan menjadi seperti tabel berikut:

Tabel 4 Nilai Pergeseran Karakter [a,b] Adalah x[i] Dan x[i+1]

brBc	A	I	M	N	S	*
A	7	7	6	7	7	7
I	7	7	6	7	7	7
M	5	7	6	7	7	7
N	7	7	6	7	7	7
S	7	7	6	7	7	7
*	7	7	6	7	7	7

Bernilai m-i (a)

Perulangan terus berlangsugan ketika nilai i=1, maka indeks berada pada posisi x[1]dan x[2] yang diisi karakter A dan N.

Rumus: m-i
Keterangan: m adalah panjang *pattern*, i adalah banyak perulangan
Penyelesaian: m-i = 5-1 = 4

Nilai pergeseran akan berubah menjadi tabel berikut:

Tabel 5 Nilai Pergeseran Karakter [a,b] Adalah x[i] Dan x[i+1]

brBc	A	I	M	N	S	*
A	7	7	6	4	7	7
I	7	7	6	7	7	7
M	5	7	6	7	7	7
N	7	7	6	7	7	7
S	7	7	6	7	7	7
*	7	7	6	7	7	7

Bernilai m-i (b)

Ketika nilai i=2, maka indeks berada pada posisi x[2] dan x[3] yang diisi karakter Ndan I.

Rumus: m-i
Keterangan: m adalah panjang *pattern*, i adalah banyak perulangan
Penyelesaian: m-i = 5-2 = 3

Sehingga nilai pergeseran akan terlihat seperti pada tabel berikut:

Tabel 6 Nilai Pergeseran Karakter [a,b] Adalah x[i] Dan x[i+1]

brBc	A	I	M	N	S	*
A	7	7	6	4	7	7
I	7	7	6	7	7	7
M	5	7	6	7	7	7
N	7	3	6	7	7	7
S	7	7	6	7	7	7
*	7	7	6	7	7	7

Bernilai m-i (c)

Ketika nilai i=3, maka indeks berada pada posisi x[3] dan x[4] yang diisi karakter Idan S.

Rumus: m-i
Keterangan: m adalah panjang *pattern*, i adalah banyak perulangan
Penyelesaian:m-i = 5-3 = 2

Sehingga nilai pergeseran akan terlihat seperti tabel berikut:

Tabel 7 Nilai Pergeseran Karakter [a,b] Adalah x[i] Dan x[i+1]

brBc	A	I	M	N	S	*
A	7	7	6	4	7	7
I	7	7	6	7	2	7
M	5	7	6	7	7	7
N	7	3	6	7	7	7
S	7	7	6	7	7	7
*	7	7	6	7	7	7

Bernilai m-i (d)

4) Tahap 4, nilai pergeseran berganti menjadi x [m-i] untuk karakter a atau karakter terakhir pada *pattern*. Apabila karakter terakhir pada *pattern* adalah S, maka brBc menjadi brBc [S][b] dan pada baris S, nilai yang dihasilkan adalah 1, karena sesuai dengan syarat algoritma berry ravindran, hasil dari nilai pergeseran x[m-1] selalu bernilai 1. Seperti pada tabel berikut:

Tabel 8 Nilai Pergeseran Karakter a Bernilai x[m-i]

brBc	A	I	M	N	S	*
A	7	7	6	4	7	7
I	7	7	6	7	2	7
M	5	7	6	7	7	7
N	7	3	6	7	7	7
S	1	1	1	1	1	1
*	7	7	6	7	7	7

Setelah semua tahap pada fase *preprocessing* selesai, maka didapat nilai pergeseranseperti berikut ini:

Tabel 9 Nilai Hasil Pergeseran Pada Fase *Preprocessing*

brBc	A	I	M	N	S	*
A	7	7	6	4	7	7
I	7	7	6	7	2	7
M	5	7	6	7	7	7
N	7	3	6	7	7	7
S	1	1	1	1	1	1
*	7	7	6	7	7	7

2. Fase Pencarian

Fase pencarian adalah melakukan pencocokan *string* dan arah kiri menuju kanan. Apabila terjadi ketidakcocokan, maka pola digeser sesuai dengan aturan Algoritma *Berry Ravindran* dapat dilihat dari 2 karakter setelah *pattern* paling kanan, lalu lihat nilainya pada tabel tahap terakhir fase *preprocessing*.

1) Langkah pertama *pattern* tidak sesuai dengan teks sehingga nilai pergeseran selanjutnyadicari dari 2 karakter setelah *pattern* paling kanan, yaitu M dan A. Pada tabel pergeseranfase *preprocessing* brBc M dan A bernilai 5, maka pola akan bergerak sejauh 5 karakterke kanan.

Tabel 10 Tabel Fase Pencarian teks (a)

T	A	M	M	A		M	A	N	I	S		T	A	G	H	N	A	A
P	M	A	N	I	S													

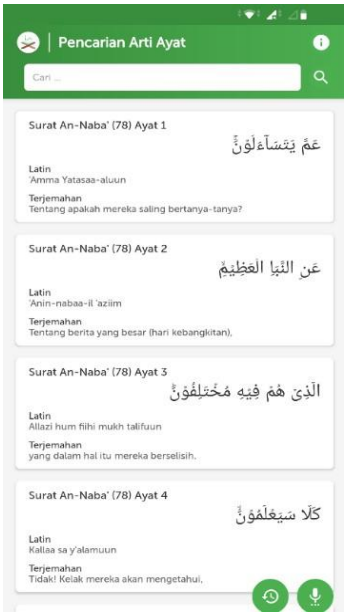
2) Pada tahap dua *pattern* sudah sesuai dengan teks, maka proses pencarian telahberakhir.

Tabel 11 Tabel Fase Pencarian teks (b)

T	A	M	M	A		M	A	N	I	S		T	A	G	H	N	A	A
P						M	A	N	I	S								

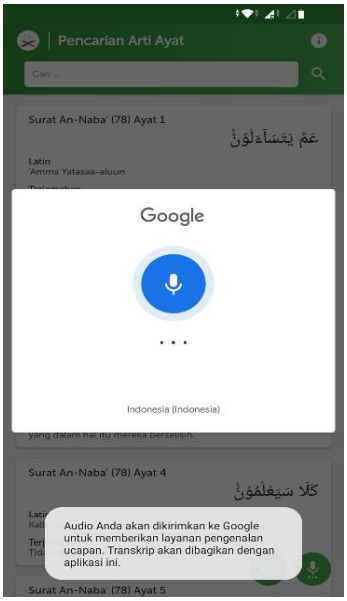
IMPLEMENTASI

Berikut hasil implementasi program keseluruhan yang telah di rancang penulis adalah Menu Utama sebagai *user interface*. Menu utama menampilkan Box Pencarian menggunakan Keyboard, Tombol Pencarian menggunakan Suara, Tombol Tentang App dan Tombol Histoy Pencarian. Berikut ini adalah tampilan Menu Utama pada Aplikasi Pencarian Arti Ayat-Ayat Al-qur'an Dengan *Speech Recognition* Berbasis Android:



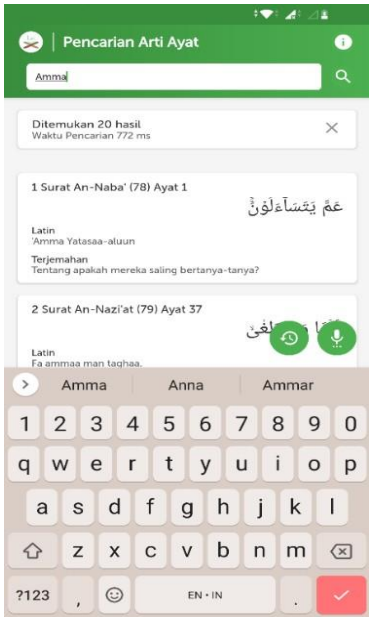
Gambar 1 Tampilan Menu Utama

Pencarian Menggunakan Suara dilakukan dengan mengklik tombol *Microphone*, lalu akan muncul Google Speech API, kemudian siswa dapat mengucapkan bagian ayat yang dicaridengan jelas dan perlahan. Berikut ini adalah tampilan Pencarian Menggunakan Suara pada Aplikasi Pencarian Arti Ayat-Ayat Al-qur'an Dengan *Speech Recognition* Berbasis Android:



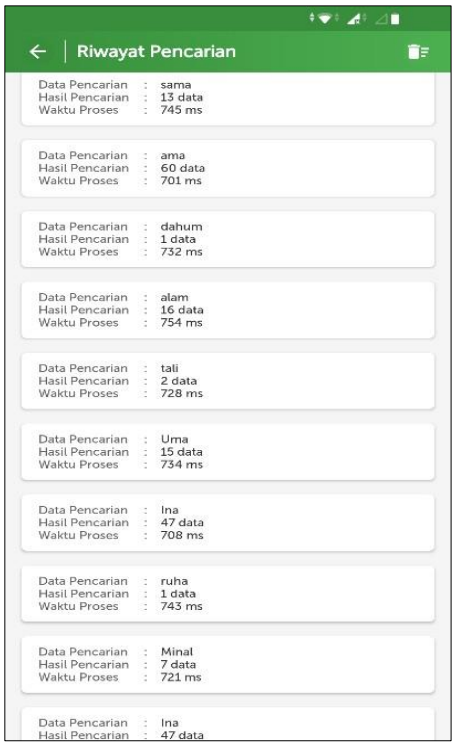
Gambar 2 Tampilan Pencarian Menggunakan Suara

Pencarian Menggunakan *Keyboard,* pada menu utama terdapat *Box* Pencarian, klik pada *Box* Pencarian kemudia akan menampilkan *Keyboard* untuk membantu mengetik pencarian kata, setelah selesai mengetikan lalu klik tombol Search untuk melakukan pencarian.Berikut ini adalah tampilan Pencarian Menggunakan *Keyboard* pada Aplikasi Pencarian Arti Ayat-Ayat Al-qur'an Dengan *Speech Recognition* Berbasis Android:



Gambar 3 Tampilan Pencarian Menggunakan *Keyboard*

Pada Menu Utama terdapat Tombol History Pencarian untuk menampilkan riwayat- riwayat pencarian yang dilakukan oleh siswa. Berikut ini adalah tampilan *History* padaAplikasi Pencarian Arti Ayat-Ayat Al-qur'an Dengan *Speech Recognition* Berbasis Android:



Gambar 4 Tampilan Daftar History Pencarian

Pengujian Pencarian Kata Menggunakan Suara dan Keyboard

Pengujian pencarian menggunakan suara dan *keyboard* menggunakan 15 kata yang berbeda dan setiap kata diulangin sebanyak 5 kali percobaan sehingga diperoleh kecocokan antara kata masukan dan kata keluaran dengan persentase akurasi ketika menggunakan suara82.6% dan ketika menggunakan *keyboard* 100%. Beberapa faktor yang mempengaruhi akurasi sistem yaitu ketepetan pengucapan dan intonasi suara.

Waktu proses pencarian tidak berpengaruh dengan pencarian menggunakan suara atau pun *keyboard*. Hasil waktu proses pencarian tergantung jumlah *database* yang ada pada sistem.Nilai rata-rata waktu proses pencarian menggunakan suara yaitu 706.7/ms dan nilai rata-rata

waktu proses pencarian menggunakan *keyboard* yaitu 704.6%. Berikut merupakan tabelPengujian pencarian menggunakan suara dan *keyboard*.

Tabel 12 Pengujian Pencarian Kata Menggunakan Suara dan *Keyboard*

NO.	KATA	SUARA		*KEYBOARD*		WAKTU PROSES PENCARIAN	
		DITEMUKAN	TIDAK DITEMUKAN	DITEMUKAN	TIDAK DITEMUKAN	SUARA	*KEYBOARD*
1.	Amma	0	5	5	0	655/ms	692/ms
2.	Alam	5	0	5	0	742/ms	715/ms
3.	Abasa	5	0	5	0	704/ms	730/ms
4.	Anin	5	0	5	0	683/ms	674/ms
5.	Anzalna	5	0	5	0	693/ms	711/ms
6	Arda	5	0	5	0	717/ms	698/ms
7.	Jannatu	3	2	5	0	669/ms	664/ms
8.	Kafa	5	0	5	0	718/ms	720/ms
9.	Minal	5	0	5	0	735/ms	686/ms
10.	Mutah	5	0	5	0	696/ms	733/ms
11.	Naumakum	4	1	5	0	717/ms	712/ms
12.	Naba	5	0	5	0	719/ms	712/ms
13.	Nama	5	0	5	0	717/ms	735/ms
14.	Summa	5	0	5	0	709/ms	682/ms
15.	Yatasa	0	5	5	0	727/ms	705/ms
		$\frac{62\times100}{75}$ = 82.6%		$\frac{75\times100}{75}$ = 100%		Rata-rata 706.7/ms	Rata-rata 704.6/ms

KESIMPULAN

Berdasarkan hasil akhir penelitian skripsi, penulis menarik beberapa kesimpulan,berikut kesimpulan yang dimaksud antara lain:

1. Penganalisisan *string matching* menggunakan Algoritma *Berry Ravindran* dan penerapanAlgoritma *Berry Ravindran* pada Aplikasi Pencarian Arti Ayat-Ayat Al-qur'an Dengan *Speech Recognition* Berbasis Android disimpulkan bahwa semakin mempermudah pencarian kata pada ayat, contohnya pada pencarian kata "Manis", setelah hasil pencariantampil dapat dilihat ditemukan 1 hasil yang cocok dengan kata "Manis" yaitu "Amma Manis Taghnaa".
2. Penerapan *speech recognition* pada Aplikasi Pencarian Arti Ayat-Ayat Al-qur'an menggunakan *Google Speech API* cukup kesulitan mendeteksi ucapan dikarena ketepatanpengucapan dan intonasi suara. Sesuai dengan tabel Pengujian pencarian menggunakan suara dan *keyboard* menggunakan 15 kata yang berbeda dan setiap kata diulangin sebanyak 5 kali percobaan sehingga diperoleh kecocokan antara kata masukan dan kata keluaran dengan persentase akurasi

ketika menggunakan suara 82.6% dan ketika menggunakan *keyboard* 100%.

3. Terbatasnya pencarian juga menggunakan suara jika melakukan pencarian ayat yang cukup panjang contohnya "akhbaaraha", memiliki huruf *double* contohnya kata "Amma"dan memiliki pengucapan yang mirip contohnya "Yatasa" saat diucapkan yang terdeteksiadalah "Ya tasya" atau "Yah tasya".

DAFTAR PUSTAKA

Akbar, A., Husodo, A. Y., & Zubaidi, A. (2019). Implementasi Google Speech API Pada Aplikasi Koreksi Hafalan Al-Quran Berbasis Android. *Jurnal Teknologi Informasi, Komputer, Dan Aplikasinya (JTIKA)*, *1*(1), 1–8.

Al Fikri, I. (2016). Aplikasi Navigasi Berbasis Perangkat Bergerak dengan Menggunakan Platform Wikitude untuk Studi Kasus Lingkungan ITS. *Jurnal Teknik ITS*, *5*(1), 48–51.

Anggraeni, E. Y., & Irviani, R. (2017). Pengantar sistem informasi. (E. Risanto, Ed.). Yogyakarta: Penerbit Andi.

Ernawati, E., Johar, A., & Setiawan, S. (2019). Implementasi Metode String Matching Untuk Pencarian Berita Utama Pada Portal Berita Berbasis Android (Studi Kasus: Harian Rakyat Bengkulu). *Pseudocode*, *6*(1), 77–82.

Fadillah, U. (2018). *Perbandingan Algoritma Berry Ravindran dan Algoritma Knuth Morris Pratt pada Aplikasi Kamus Gizi Berbasis Web*.

Huda, Miftahul. 2019. *Algoritma Data Mining : Analisis Data Dengan Komputer*. EBOOK. Yogyakarta

Mukaromah, H., & Amelia, K. R. (2019). *Perancangan Aplikasi Penjualan Tapis Lampung Berbasis Android*.

Nada, Q., Ridhuandi, C., Santoso, P., & Apriyanto, D. (2019). *Speech Recognition dengan Hidden Markov Model untuk*. *5*(1), 19–26.

Siburian, E. O. (2017). *Implementasi dan Perbandingan Algoritma Berry-Ravindran dan Zhu-Takaoka pada Aplikasi Kamus Bahasa Indonesia – Batak Toba*